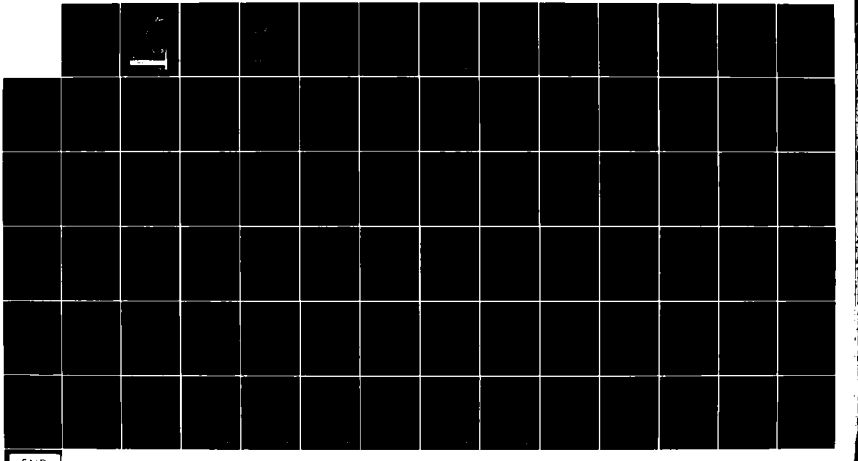
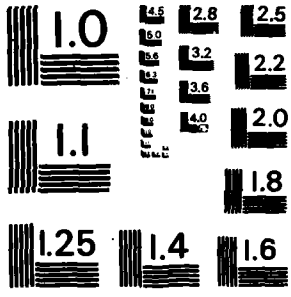
AD-A133 644 DESIGN FOR AN ADVANCED RED AGENT FOR THE RAND STRATEGY ASSESSMENT CENTER(U) RAND CORP SANTA MONICA CA R STEEB ET AL. MAY 83 RAND/R-2977-DNA DNA001-80-C-0298 1/1

UNCLASSIFIED F/G 15/7 NL



END
DATE FILMED
10 83
DTIC



MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS - 1963 - A

Design for an Advanced Red Agent for the Rand Strategy Assessment Center

Randall Steeb, James Gillogly

Rand

The research described in this report was sponsored by the Defense Nuclear Agency under Contract No. DNA001-80-C-0298.

Library of Congress Cataloging in Publication Data

Steeb, Randall, 1946-
Design for an advanced Red Agent for the Rand Strategy Assessment Center.

"Prepared for the Defense Nuclear Agency."
"June 1983."
"R-2977-DNA."
1. Red Agent (Computer war game) 2. Rand Strategy Assessment Center. I. Gillogly, James. II. Rand Corporation. III. United States. Defense Nuclear Agency. IV. Title.
U310.S83 1983 355.4'8'02854 83-9773
ISBN 0-8330-0505-7

Published by The Rand Corporation

R-2977-DNA

Design for an Advanced Red Agent for the Rand Strategy Assessment Center

Randall Steeb, James Gillogly

May 1983

Prepared for the
Defense Nuclear Agency

35th Year

Rand
SANTA MONICA, CA. 90406

PREFACE

This report examines technical options for modeling behavior of the Soviet Union in automated war games for the Rand Strategy Assessment Center (RSAC). It proposes an evolutionary development program for an advanced Soviet model (a "Mark III Red Agent"). The RSAC is an ambitious multi-year research effort to develop wargame-based methods for improving the quality of strategy analysis for conflicts up to and including general and prolonged war. It is supported by the Director of Net Assessment in the Office of the Secretary of Defense, and by the Defense Nuclear Agency, under Contract DNA 001-80-C-0298.

The cutoff time for this report was January 1983. Since that date, the RSAC has made considerable progress toward building an advanced Red Agent. There have been several changes in the technical design, but the principal ideas laid out here continue to apply.

Comments and inquiries are welcome and should be addressed to the authors or to Dr. Paul K. Davis, Director of the Center. Related research is reported in the following Rand publications:

Paul K. Davis and James A. Winnefeld, *The Rand Strategy Assessment Center: An Overview and Interim Conclusions About Utility and Development Options*, The Rand Corporation, R-2945-DNA, March 1983.

William L. Schwabe and Lewis M. Jamison, *A Rule-Based Policy-Level Model of Nonsuperpower Behavior in Strategic Conflicts*, The Rand Corporation, R-2962-DNA, December 1982.

Carl H. Builder, *Toward a Calculus of Scenarios*, The Rand Corporation, N-1855-DNA, January 1983.

Paul K. Davis, *Concepts for Improving the Military Content of Automated War Games*, The Rand Corporation, P-6830, November 1982.

SUMMARY

This report describes Rand's proposed approach for building an advanced "Red Agent" computer model to describe Soviet behavior in the context of an automated war game. Rand's Strategy Assessment Center (RSAC) is developing automated war games as a potentially revolutionary tool for strategy analysis.[1]

The advanced (Mark III) Red Agent described in this report will use and extend state-of-the-art artificial intelligence techniques to produce an extremely flexible model able to reflect the best-estimate and contrary-view concepts on likely Soviet political-military behavior in major superpower conflicts.

The role of the Mark III Red Agent will be quite demanding. The Red Agent will, for example:

- Make decisions at the level of strategy and operational art for allocation and employment of forces in conflicts up to and including general and prolonged nuclear war—i.e., decisions involving disparate force types in many different regions.
- Base decisions not only on the current state of the world, but also on projected actions by the United States and nonsuperpower countries, and on projected outcomes of ongoing or anticipated battles.
- Adjust its projections as conflict continues and it learns more about actual U.S. strategy and other factors.

Rand's approach to these matters places heavy emphasis on qualitative factors that go well beyond the technical criteria used in standard models. For example, the Red Agent's decisions must reflect judgments about U.S. and allied intentions and will. These judgments are based both on Agent predispositions (as inputs) and on objective indicators arising in the course of the war game (e.g., U.S. decision times, decisiveness, and success in rapid mobilization).

There are two aspects to the development of an advanced Red Agent. The first is technical: The model must interact effectively with computational models representing the United States and other countries; its decisions must be readily understandable (i.e., the underlying rules should be able to be recalled in interactive operations); and it must be able to readily accept changes in individual decision rules or even changes in patterns of Soviet behavior. Indeed, the model must permit occasional human intervention or even free-play by hu-

man teams for many decisions. Second, the model's rules must reflect the best information available on Soviet doctrine, Soviet-style strategy options, and other Soviet decisionmaking characteristics. In fact, different sets of rules will be necessary for different estimates of Soviet character.

After reviewing the artificial intelligence techniques used in other ambitious applications, and after substantial debate about how best to accommodate expert military judgment on war fighting at the level of strategy and operational art, we have settled on a hybrid approach combining and extending the techniques of scripting and goal-directed search. In our context, "scripts" are closely related to militarily credible war plans. Military experts will prepare a range of possible Soviet high-level war plans and identify as many plausible branches as possible—including controversial ones. These will then become the basic framework for the Red Agent's scripts. These scripts will take a building-block form that can be used in a variety of circumstances. This requires the Red Agent to be able to tune the scripts to the occasion (e.g., allocate forces appropriately, time actions, etc.) while keeping the underlying goals firmly in mind. Some of this tuning will be inherent in the scripts, in the form of condition-action rules triggered by the situation encountered, and some tuning will be accomplished through the use of goal-directed search techniques. For example, if essential objectives are not satisfied in the projected outcomes, the search techniques will be used to modify the script actions systematically.

The advanced Red Agent will also have a three-level system structure to better distinguish the types of decisions ordinarily made at the national-command, area or functional command, and tactical levels. This compartmentalization of knowledge in the system should facilitate expert interaction during rule entry. Experts entering information will have to consider only those situations and rules relevant to their specialty areas. The multilevel system should also be instrumental in capturing U.S. and Soviet asymmetries in decisionmaking.

We plan to develop the Red Agent in an evolutionary manner. We will first demonstrate a prototype system (which we term the Mark III.1) with a limited number of scripts applicable to a single escalatory conflict involving Southwest Asia and Western Europe. This initial system will be highly interactive, with frequent adjustment of rules until effective and credible decisions are produced. The behavior of the Blue Agent, the Force Agent, and the Scenario Agent will be highly constrained. In the mature Mark III system, the other agents may be unconstrained, or in certain modes, they may be free-played by human operators. During analysis functions with the mature system, the human operator will act more as manager than participant. The

system design will allow the operator to define the conflict situation, monitor the system's choices, intervene when problems arise, query the logic behind choices, and request various types of analysis.

ACKNOWLEDGMENTS

Many members of the Rand staff helped define the Mark III Red Agent design described here. William Jones provided examples of strategic and tactical decisionmaking, gave numerous recommendations on system structure, and provided comments on our drafts. Roberta Allen devised several scenarios used for hypothesizing and testing Red Agent functions. Jonathan Cave, William Schwabe, Robert Levine, and Philip Klahr provided inputs on some of the theoretical aspects of the approach. Paul Davis and James Winnefeld provided invaluable advice for both the structure and content of this report. Any inconsistencies or shortcomings that remain are the responsibilities of the authors.

CONTENTS

Appendix

FIGURES

TABLES

I. INTRODUCTION

This report examines design options and describes a proposed configuration for an advanced Mark III Red Agent in the Rand Strategic Assessment Center (RSAC). This Red Agent, responsible for modeling Soviet strategic decisionmaking, has a complex and multifaceted role. It must orchestrate strategic and tactical planning for all Soviet forces in all theaters. It must manage intelligence gathering, execute actions, and monitor outcomes, while adjusting its behavior when new situations arise. The Red Agent must produce a trace of its actions and processes and facilitate human interaction. Also, the design for the Red Agent should be applicable to the complementary Blue Agent (modeling U.S. strategic decisionmaking). Thus the structure must be sufficiently flexible to reflect highly varied decisionmaking processes and behavior.

The Red Agent is one of the major components of the RSAC's automated wargaming system, a system intended to facilitate development, testing, and analysis of strategic and tactical military plans. This system attempts to combine the analytic virtues of computer simulation and modeling with the richness of traditional political-military war games.[1] During initial development phases of the RSAC, Rand demonstrated the feasibility of replacing the players in the traditional war game with programmed automata—automated computer-based problem solvers capable of simulating the processes of strategic decisionmaking. We refer to these automata as the Red Agent (representing Soviet behavior), the Blue Agent (representing U.S. behavior), and the Scenario Agent (representing the behavior of all other countries). Additionally, the system includes a Force Agent responsible for outcome calculation—i.e., estimating the status of all forces and targets and describing conflict outcomes—and a Systems Monitor for keeping track of timing, moves, and historical records (the Scenario and Force Agents and the Systems Monitor are described in Refs. 2, 3, and 4). In this report, we describe a proposed new structure for the Red Agent and specify some of its interactions with the other system automata.

Our proposed design for the Mark III Red Agent will require several advances over the current state of the art in knowledge engineering and planning systems. The problem domain, multitheater strategic and tactical decisionmaking, has to our knowledge experienced no attempts at full-scale automation. Compounding this, the RSAC system

has special requirements of transparency, repeatability, rapid turnaround, and interactive ease. To achieve these objectives, we propose a hybrid approach, new to automated planning, that combines the artificial intelligence techniques of scripting and goal-directed search. Scripting involves specifying time-tagged sequences of actions to achieve specific goals.[5, 6, 7] The Red Agent selects different paths in a script and different actions depending on the situational conditions encountered. The proposed system then performs a simulation-based look-ahead of the scripted actions and uses goal-directed search techniques to modify the actions when goals are not achieved.

We outline a development cycle for the Mark III system that begins with an initial prototype system useful in a limited domain. This initial system will rely heavily on human input of key decision parameters. Following a period of expanding and refining the system rules, we plan to implement a fully automated system capable of modeling and analyzing many types of strategic conflict (up to and including prolonged general war).

CHOICE OF A TECHNIQUE FOR RED AGENT AUTOMATION

During the course of our development of the Red Agent, we examined several established methodologies that might be applicable to automated strategy-level war gaming. These included search techniques,[8, 9] production systems,[10, 11, 12] script-based approaches, [6, 7] decision analysis methods,[13, 14] and pattern classification techniques.[8, 15] The choice of method is extremely important, as the task is on the order of such major artificial intelligence systems as MYCIN,[16] DENDRAL,[17] or HEARSAY.[18] Appendix A discusses the relative capabilities of each of the methodologies for automation of the Red Agent and rationalizes our hybrid choice. The criteria we used for selecting a technique are given below:

- *Multilevel Structure.* The system design should reflect several organizational levels, each with its own structure, responsibility, and links to the other levels.
- *Transparency.* The system's actions should be easily understandable. This requires maintaining traces of all decisions and showing the instant situation, the options considered, the reasons for choosing each action, the time the action was executed, and the outcomes. Transparency also requires easy interpretation and modification of agent rules by experts.

- *Repeatability*. The system must act consistently, replicating a planning action when an identical situation is encountered. This is essential for comparing alternative strategies.
- *Credibility*. The system must act in a manner consonant with judgments by experts on Soviet military style.[1] Although the system is not intended to precisely simulate Politburo decisionmaking, its choices should accord with those made by Soviet strategists in actual past conflicts (e.g., Hungary in 1956). At the same time, the system must respond in a reasonable manner to totally new situations. It must be able to generalize its behavior using some form of similarity measure to known situations.
- *Gaming Capability*. The system should explicitly model the opponent's actions over several cycles (of game play). We expect that nongaming decision rules attempting simply to match the current situation without modeling opponent responses will fare poorly compared with a gaming problem solver. A balance must be struck, however, because real-world decisionmakers (or organizations) are highly limited in the amount of opponent modeling they can perform.
- *Speed*. The system must respond within minutes of being presented with a problem situation. Because of the large number of action specifications and look-aheads present with each move (on the order of dozens to hundreds), this will necessitate efficient communications between system elements, rapid computations by Force and Scenario, and early pruning of nonviable options.
- *Treatment of Uncertainty*. Although the responses of the system must be replicable, the system must model an uncertain environment. The status of enemy forces, the opponent "will," and the political climate must often be estimated from unreliable, outdated, and conflicting information. In its mature version, the Red Agent must thus represent and update uncertainty estimates.
- *Ability to Event-Step*. Time must be explicitly represented in the modeling process, but stepping the system forward in short, discrete time increments may be inefficient. Events occur sporadically in a strategic conflict. Stepping the system forward by events may reduce the processing load considerably compared with a time-stepped process.[21] Event-stepping may, however, require extensive calculations to determine the quiescent periods between steps.[22]

[1] A great deal of information is becoming available on this subject, some in the form of doctrine, and some in the form of exercise behavior (see Refs. 19 and 20).

- *Distributed Commands.* At the area or functional level, the theaters may act quite independently. Each theater (e.g., Western Europe, Thrace) receives its commands from the top-level strategic planner, then invokes the needed actions based on the local situation. If problems develop, the theater command may send messages horizontally to his analogues, or vertically to the strategic planner, depending on the command structure and the situation. Asymmetries between U.S. and Soviet decisionmaking may be modeled by changing the responsibilities at each level and the rules for communicating between levels.
- *Interactive Human Interface.* Among the RSAC objectives are the support of interactive sensitivity analyses and the modeling of hypothetical situations input during a run. The system manager must be able to interactively input modifications to key parameters and view the results graphically or textually. Experts should also be able to interrogate the system regarding reasons for action selection.

From the viewpoint of computer science, the most important of the above technical requirements for the Red Agent (and for the RSAC system as a whole) appear to be reasoning capability, transparency, response time, and ease of interaction. These characteristics are essential to the task of systematically generating and comparing strategic planning options. Our analysis of the several approaches (see Appendix A) indicates that a generalization of scripting is the most effective method for for representing the complex, time-sequenced actions of the Red Agent. Scripts efficiently and transparently show the progression of action selection and branching under changing conditions.[6, 7] Scripts maintain a context, so that matching of actions to situations is much less cumbersome than in other methods, and move-to-move consistency is improved. Still, we need to augment the scripting approach using such techniques as condition-action rules to adjust script behavior in changing situations and goal-directed search to modify actions when projections indicate that goals will not be met.

PROPOSED APPROACH

Our planned approach for the Red Agent uses branched scripts—sequences of connected, time-tagged actions with branching points. Experts first prepare such scripts for each major strategic option and area, such as an invasion of Afghanistan, an invasion of NATO Europe, or an initiation of intercontinental nuclear war. The Red

Agent follows the actions specified in the script, filling in details and taking branch options as they are encountered. During this process, the script specifies goals to be achieved by certain times. The Red Agent uses feedback mechanisms to modify script actions until these goals are met. For example, the Red Agent may reach a script point calling for use of strategic nuclear weapons. In look-ahead, Red considers a worldwide countermilitary strike, with the goal of limiting Blue's retaliatory capability against military targets to a given level. Depending on such game indicators as Blue state of alert, Red's model of Blue may indicate a Blue launch-under-attack that would partially thwart Red goals. This might cause Red to backtrack and use an additional strategy of blinding U.S. early warning systems.

Our proposed Red Agent has three interacting levels to execute these scripts, much like the hierarchies found in military chains of command. The top level, corresponding roughly to the national command level (NCL), examines the situation and selects the script best suited to the situation and the Soviet goals. The middle or area command level (ACL) is composed of separate planners for each functional area or theater. The ACL planners develop and refine the war plans for their area of responsibility (subject to objectives, constraints, and interface requirements). The lowest level is the TCL or tactical command level. This level applies appropriate tactical doctrine to the actions and interacts with the Force Agent to project outcomes and execute the actions.

The data flows among the three levels are arranged in the following manner. After the NCL selects a script, it allocates resources (troops, armor, equipment) to the various ACL planners and passes them the actions and time-tagged goals specific to their areas. Each ACL planner refines the actions passed to it, calculating action times and filling in force levels and action types. As the action times are reached, the actions are then passed to the TCL, where the outcomes are simulated by Force Operations and the results entered provisionally into the Red Agent database. The three levels are not completely separate, of course. They all reside in the same program as distinct sets of rules called by the Red Agent scheduler, and they all have access to the agent database that describes Red's beliefs about the current situation.

During the script simulation process, the proposed Red Agent not only fills in its own actions but also predicts the responses that Blue will make. Using its estimate of Blue strategy, Red fits a Blue script at each update of the situation and enters its estimate of upcoming Blue actions provisionally into its world model. At the next Red move point, the Red Agent continues its specification of its own actions in the simulated future. The Red Agent schedules its various activities

through use of a number of different tests—branch tests, continuation tests, move tests, and goal evaluations.

The Red Agent operates in two modes—look-ahead and actual time operation. During look-ahead, the Red Agent simulates its own and Blue's actions, predicts outcomes using the Force Agent, and adjusts its behavior by modifying earlier actions. This is all done with time frozen (no actual game time elapses). During actual time operation, the Red Agent executes its actions, observes the Blue responses, and if necessary initiates a new look-ahead cycle.

The scripts themselves act as building blocks, and may be linked together in a scenario. Different scripts may be used for different areas of conflict such as Western Europe and the Pacific, and for different levels of escalation such as conventional warfare, theater nuclear war, and strategic nuclear war. Each script has its own goals and action sequences, along with rules for coordinating with the actions of other scripts.

SYSTEM INTEGRATION

Our design for the Red Agent will require augmentations in the capabilities of the other agents and in the human interface. The Scenario Agent, responsible for all nonsuperpower military and political actions, must be quickly able to predict responses to each given projected situation. It may be necessary, in fact, to encapsulate some of the scenario rules into the Red Agent to ensure rapid response. Force operations similarly must act quickly, calculating the results of many projected engagements. The Force Agent must also refine actions by applying local tactical doctrine and checking for satisfaction of critical assumptions. The human interface, finally, will have to be enhanced by including provisions for graphical and textual display of progress along the script, by allowing operator input of rules, parameter changes, and queries, and by providing graphic display of sensitivity analyses of key parameters and assumptions. We will describe each of these characteristics in detail in later sections.

DEVELOPMENTAL PLANS

We expect to implement a very limited version of the Mark III Red Agent late in FY83. This initial system will have a limited number of scripts applicable to only one or two simplified cases (e.g., territorial acquisition options in the Persian Gulf area and strategic nuclear

war). We will simplify the simulation demands in this system by making the possible Blue responses known to Red, by specifying all move and evaluation points, and by precalculating many of the required Force and Scenario inputs. We will use this simplified system in an interactive testing mode—systematically assessing the quality of the Red choices and adjusting the rules and database entries on-line until the system produces effective strategic plans. The main experimental variables in this exploratory phase will be the initial conditions—force levels, political situations, alliance structures, and agent character.

We will schedule more traditional experimental studies following the shakedown period, involving complex multitheater situations, escalation through different weapon levels, and unconstrained Blue responses. We expect the system to evolve forward through expert input of new data, action rules, and control parameters.

II. STRATEGIC PLANNING AND THE RSAC

The Rand Strategy Assessment Center is a system built around an automated political-military game.[1] The intent of this automated system is to retain the structure of a traditional political-military game but provide the ability to replace the free-play teams with programmed agents. Such a structure allows much of the rich contextual complexity of free-form gaming but, by replacing humans with automated programs, has much of the replicability and rigor of analytic modeling.

The RSAC will provide a tool for analyzing Soviet decisionmaking, developing U.S. strategic plans, and testing the robustness of those plans under different assumptions. It may be used to evaluate alternative force structures, determine needed acquisitions, and guide deployments. Accordingly, the Red Agent must accurately simulate many aspects of Soviet behavior during military conflict situations. In the early phases of system development such simulation was done on a shared control basis, with teams of human experts accepting machine recommendations and selecting appropriate options.[1, 2] In the Mark III system, we will more fully automate the Red Agent's decisionmaking and enhance the facilities for close human interaction.

GAME STRUCTURE

The essential structure of the RSAC political-military gaming system is shown in Fig. 1. The Blue team plays the role of the U.S. national-level and theater-level decisionmakers, and the Red team plays comparable roles for the USSR. The Force Agent is responsible for force calculation and execution of military actions called for by the other agents. The Scenario Agent is responsible for generating all nonsuperpower political information and actions. In effect, Scenario portrays the worldwide political situation to the adversaries. The Systems Monitor coordinates the game communications, maintains databases, and keeps the game clock and records.

Moves in a game exercise are performed sequentially. After each adversary (Blue or Red) move, the Systems Monitor advances the game time and game step, Force Operations updates the military situation, and Scenario updates the political situation. This procedure sequences the control functions and allows step-by-step recording of exercise events for subsequent analysis.

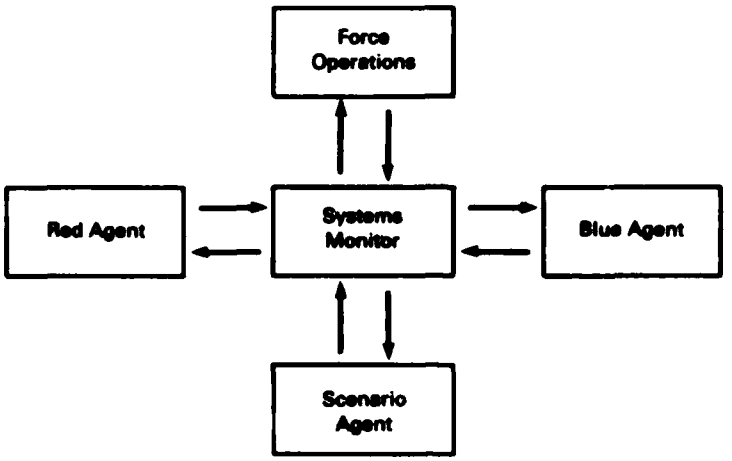


Fig. 1—RSAC system structure

SYSTEM DEVELOPMENT

The development of the RSAC is now in its second phase. Mark I resulted in the demonstration of a rudimentary system that proved the feasibility of automating one or more of the components of the traditional political-military game. This system consisted of a combination of automated pattern matching algorithms augmented by expert human input. The Mark II system, which recently completed testing, enhanced the automated planning functions of the Mark I system by including rule-based refinement of actions and by adding a tactical decision level. The Mark II system also increased the capabilities of the Scenario Agent to produce responses automatically and incorporated some marginal improvements in the Force Agent's capabilities. The Mark III system, now in development, will further automate the strategic planning process by specifying actions in the form of scripts, by systematically modifying actions to achieve specific goals, and by utilizing a three-level structure that more closely approximates the dynamics of military decisionmaking.

In the Mark I system, some of the system operations were simulated using human input so that resources could be concentrated on automating the Red Agent. Choice of actions in the Red Agent involved a computer implementation of a production system, using the If-Then format. The if-conditions were specified in terms of some 15 different situation descriptors. The Red database comprised a set of several

hundred such situation descriptions with action instructions associated with each one. The Red control system would invoke a pattern matching process, finding the best match to the situation descriptors. The four best fits (in the sense of a weighted distance measure) were shown along with their action instructions to the Red operator who would then implement the best fit action considered appropriate. This procedure worked well in a demonstration exercise, showing reasonable choices of high-level strategic actions.

The Mark II system improved on the initial version by incorporating rule-based systems supporting the Red, Blue, and Scenario Agents. These systems specified action details based on matches of the projected situational conditions to the rule antecedents. Some of this rule-based decisionmaking required extensive look-ahead calculations from Force. Unfortunately, this approach was limited as it had little representation of past or future. The only factor assuring continuity was use of antecedent conditions such as, "If you are on script A and this condition applies, then take action A3." A major role of the Mark II system was as a testbed for some of the Mark III features. The rules for action specification were roughly equivalent to the slot specification rules to be used in the Mark III system. The Mark II system also compartmentalized some of its functions into two decisionmaking levels, presaging the three-level organization of the Mark III system.

The Mark III system is targeted to be a mature automated system for developing and testing strategic planning options. The human operator will act as more of a manager than a participant, defining the conflict situation, monitoring the system, intervening when problems arise, querying the logic behind choices, and requesting analyses. The task for the automated system is fairly difficult, as the Red Agent must plan and coordinate all Red military and political activities, allocate limited resources across different theaters, manage communications with all affected countries, predict Blue responses, and maintain a dynamic model of the developing worldwide situation. The cost of this greater power and depth of analysis may be a loss in speed. The Mark III system will require much faster responses from Force and Scenario than the previous systems and will have to perform its look-ahead and planning operations much more efficiently.

III. PROPOSED SCRIPT-BASED SYSTEM

We selected a branched-script approach for automation of the Red and Blue Agents, as this methodology best satisfies the system requirements outlined in Sec. I. Appendix A gives a detailed comparison of the applicability of this and the other approaches we considered. The branched-script approach uses time-tagged actions with branching points.[6] However, the traditional use of scripts as immutable sequences of actions is broadened in our work to include modification of actions when goals are not met. The Red Agent first selects the script itself according to specific situational criteria. Condition-action rules and algebraic calculations then specify the variable elements of the script as it unfolds over time. We will elucidate all of these script application characteristics below in a component-by-component description of the proposed system. Section IV will then give a more complete view of the methodology by tracing an example scenario based on a hypothetical strategic-nuclear conflict.

MULTILEVEL STRUCTURE

Figure 2 shows the three interacting levels of the proposed Red Agent problem solver. The three levels correspond to (1) national policymaking, (2) theater military command in each function or area, and (3) tactical force application at the operations level. At the topmost level, the national command level (NCL) is concerned primarily with the best analytic war plan to use in the current situation.[1] This analytic war plan is embodied in a set of coordinated scripts chosen for the various theaters along with goals that must be achieved by specified times. A script is a segment of the analytic war plan, fleshed out to include tactical as well as strategic actions.

The second level, the area command level (ACL), is where most of the action specification activities take place. This level fills out the scripts according to the situation in each theater and repeatedly requests simulations of its choices by the third level (the tactical command level or TCL). At key times, the ACL makes an evaluation of the progress achieved. If the script does not sufficiently achieve the goals in the simulated future, the ACL makes a new attempt to fill

[1]The term analytic is used here because an analytic war plan contains contingency plans for a wide range of possible situations and is structured to accommodate the requirements of analysis supported by automated gaming.

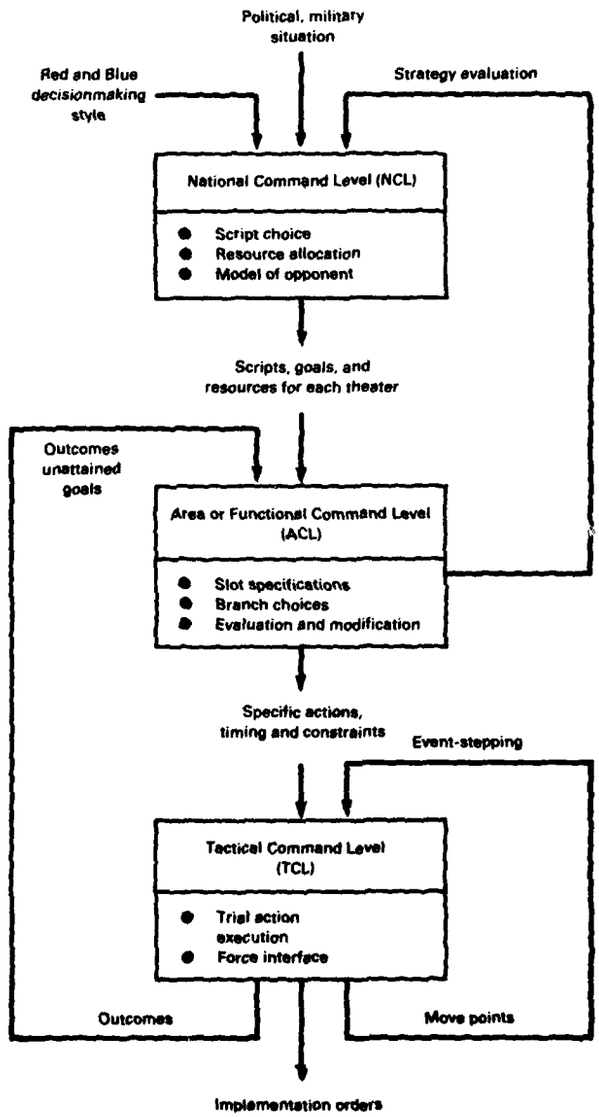


Fig. 2—Multilevel structure of the Red Agent

out the script. The new attempt may involve new times of application of actions, augmentation of force levels, or specification of new actions. If none of the attempts to modify the script achieves the goals, then the NCL reallocates resources or tries a new script. If the script satisfies the goals in look-ahead, the NCL accepts it and the script becomes an operations plan.

The NCL also uses scripts to predict the actions of the adversary Blue Agent. The NCL examines the projected situation following its own actions, compares the situation with triggering conditions for Blue responses, and activates one or several Blue scripts. The NCL simulates forward the Blue scripts just as it does the Red scripts, and it makes plans for each possible Blue response.

The Red Agent's multilevel structure is advantageous, as it allows us to partition knowledge into discrete pieces amenable for entry by experts in the area or function addressed. The rules for NCL planning, ACL action specification, and TCL force application are each separate, making operations modular and transparent. The multilevel structure results in processing efficiency because the program searches a problem-specific subset of rules only during planning. The input of rules by experts is also simplified as the experts need to consider only a single level. Finally, this structure appears to correspond well to existing systems for strategic and tactical planning, paralleling some of the forms of feedback running between military command levels. We should be able to capture the asymmetries between Red and Blue decisionmaking using different versions of the same multilevel structure.

SCRIPT ELEMENTS

A script is a sequence of one Agent's actions and branching options. The actions and branching options are independent of the Agent's character (risk-taking, opportunistic, risk-averse, etc., which we term the choice of "Ivan" for the Red Agent). That is, the same options are available to any Ivan. (The Blue Agent is "Sam.") However, the goals for the script and the rules for refining actions and choosing branching options are highly dependent on the Agent's character.

A branch is a decision point at which one of several distinct sequences of actions must be chosen. The sequences may be different tracks within a script or may involve a jump to a different script entirely.

An action itself is a detailed description of how and when some single military or political act must be performed. The action is de-

fined in terms of a constant "body" and some variable "slots," which specify action details (timing, force levels, location) that depend on the situational conditions present when the action is activated. Successful execution of the actions depends on the accomplishment of goals—essential, time-tagged objectives associated with the script.

At the top (strategic) level, a script is equivalent to a portion of an analytic war plan (AWP), which describes all possible strategic events in a military option. In an invade and control option, for example, the AWP may describe all events from the conventional attack to a nuclear escalation. Each of the major strategic sequences (conventional attack, theater nuclear preemption, strategic nuclear war) may be a separate script, as each can be a distinct phase, involving different forces, rules, and goals. The scripts define these actions down to the functional command and gross tactical level, providing much more detail than an AWP. We can consider an AWP to be a linking of the strategic elements of one or more scripts.

In many ways, we can view the scripts themselves as building blocks for strategic planning. We can define scripts for each major situation and option and string them together to produce plans. The "chunk" size here is important. If we define very small scripts, corresponding to short sequences of actions in a local area, then we may have to devote large proportions of these scripts to interfacing rules—rules for coordinating with many other small scripts. However, if we define very large scripts, able to coordinate actions over many theaters and over many strategic events, then we have to produce a large, complex script for each different situation. A balance must be struck between these two extremes.

In general, to select and fill out a script we need to delineate the following information:

- *Agent character.* The form of Agent decisionmaking, such as risk-prone and risk-averse, defines the specific set of criteria for selection and execution of a script. Although the overall structure of a script (actions, branches, etc.) is the same for all Ivans, the rules for invoking and applying those actions are specific to the different Ivans.
- *Criteria necessary for activation of the script.* These are the situational conditions used by the NCL to evaluate the applicability of a particular script. The criteria may be essential (necessary for choice of a script) or optional (contributing to the value of a script but not necessary).
- *Allocation of functions and resources to each area or theater.* The NCL partitions the script into portions specific to each area or theater command. The NCL passes to the areas or

theaters the goals, resources, constraints, and actions specific to their operations. The area or theater commands consult their rules to choose branch options, refine actions, and request projections from Force and Scenario.

- *Slots for refinement of the actions.* The actions passed the ACL are skeletal in nature. Slots for specific forces levels, tactics, and timing are filled in by the ACL according to the situation present when the action is reached during look-ahead. The ACL uses condition-action rules and algebraic calculations for this function.
- *Move, branch, and continuity tests.* These are conditional tests to determine if control should be transferred between the different agents (Red, Blue, and Scenario), which of several tracks in the script should be followed, or if the current script should be terminated and a new one begun.
- *Evaluation and termination tests.* These are tests to determine if the time-tagged goals passed to the ACL have been successfully accomplished by the script actions and if sufficient information has been collected to terminate the look-ahead. If the script performance is unacceptable, the system uses mappings between unfulfilled goals and specific actions to invoke alternative actions.

For illustration of the functions, we will use a theater conflict scenario (Appendix D describes this example scenario in detail). Briefly, the scenario describes one complete decision cycle and begins with Red's move 30 days before (D − 30) an attempted Southwest Asia invasion. At this point Red is mobilizing its forces and must determine whether to invade and occupy Northern Iran only, all of Iran, or all of the Persian Gulf. Red decision hinges on whether conditions assure an easy conventional force victory and low likelihood of Blue preemption. Red tests prospects for conventional victory by looking at the force ratio, Blue's reinforcement time, and expected Iranian resistance. Red determines that its position is sufficiently favorable to attempt a takeover of all of Iran, but not strong enough to try for the entire Persian Gulf. Blue responds with a deployment of the Rapid Deployment Force (RDF) but is unable to repel the attack.

Agent Character

Currently, the Red Agent is envisioned to take one of several agent characters or "Ivans," such as adventurist, conservative, and opportunistic. These characters are captured in distinct alternative sets of

rules, which determine the criteria used to select among scripts and among branches in a script. For example, the adventurist Ivan might choose to invade all of the Persian Gulf if a majority of the NATO forces are already committed to a conflict outside that theater. The Ivan in the example scenario is risk-averse. It will attack if the situation appears highly advantageous to Red and little chance of escalation is present. The rules for a particular Ivan also prescribe the intermediate time-tagged goals to be accomplished, specify the criteria for slot specification, and determine the possible Blue responses.

In several ways, the Ivans value scripts differently according to implicit goals. Each Ivan represents a set of national goals, such as preservation of the Soviet Union as a world power, reduction of U.S. military capabilities, and expansion of Soviet influence in the third world. The different Ivans have different values for the goals associated with each script (occupy the Persian Gulf, control Western Europe, etc.) and set different criteria for evaluation of a particular script. The criteria reflect the importance of achievement of the script goals and the risks of failure to the Ivan.

Selection Criteria

In the Mark I and Mark II versions, the systems chose actions using a weighted distance measure that measures closeness to a situation vector having some 15 to 25 dimensions. The problem with using this type of scheme for script selection is that the attributes are seldom additive. Some attributes may be essential (minimum number of warheads, necessary overflight rights, etc.) and others may simply contribute to the value of an option. In essence, some of the attributes are multiplicative and some are additive. In the Mark III Red Agent, we plan to use a two-part system for handling these distinctions. The NCL tests the essential criteria (expressed as yes/no responses) to initially cull down the set of script options. If only one script remains, the NCL invokes it. If several scripts remain as candidates, the NCL uses the contributing criteria to select among them. The contributing criteria may be either binary (present/not present) or interval scaled, and the NCL uses a weighted aggregate of the criteria for selection of the script (see Appendix C for a more complete description of this process).

Occasionally, the Red Agent may not be able to evaluate a criterion (either essential or optional) at the time of the script or branch choice. If the criterion's satisfaction makes the option the favored one, the NCL or ACL may provisionally assume the criteria to be satisfied and test it later in the simulated future. The NCL will tag the choice in

the database as having an assumption. In our example, the NCL does not know if Blue reinforcements will arrive by a certain time but assumes they will. The Red Agent will follow the script in simulation until that time is reached and the assumption tested. If the assumption turns out to be false, then the system must backtrack to the original branch or action where the assumption was made.

Allocation of Functions and Resources to the ACL

The NCL makes the handoff to the ACL by assigning goals, action sets, and resources to the different area commands in the ACL (Southwest Asia and Western Europe in the example). Each area command has its own rules for subsequently refining the actions and each has access to the situation estimate in the Red database. The NCL distributes the goals and actions to the area commands on the basis of location or function tags.

The NCL allocates resources to the different areas using two special sets of rules. The first set roughly calculates the equipment and forces required for achieving the goals set for each area. The calculation may require inquiries from NCL to Force Operations. Many theaters or areas will already have satisfactory commitments of forces. The NCL uses the second set of allocation rules to prioritize the various areas by importance and, if necessary, to pull optional forces from adjoining areas. This procedure services the most important areas first. For example, in the theater conflict scenario, the NCL must allocate Red forces to two theaters: SWAsia and Western Europe. SWAsia is the predominant theater. Red must first allocate forces to SWAsia to fulfill the requirements of the script. If the primary theater needs more forces, the NCL invokes such rules as the following:

- If the ratio of Red to Blue Amored Division Equivalents (ADEs) in the Persian Gulf is between 1.5:1 and 2:1, then move 30 percent of the ADEs from the Western European theater to the Persian Gulf.

They will be drawn from the surrounding theaters using rules such as:

- It is possible to move forces from a less important theater to a more important theater if the remaining forces maintain a ratio of at least 1:1.

These allocations are an initial attempt to distribute forces. The NCL may reallocate forces if the goals for a given theater are not accomplished in look-ahead.

Slot Specification

The actions passed from the NCL to the ACL are skeletal in form. The NCL normally specifies the action type but leaves the force level, timing, and location as slots to be filled out by the ACL. The various area commands cycle through each slot in the script according to the time of execution of the tasks (earliest first). The area commands derive the times either from time tags (perform at 0300) or enabling conditions (perform after securing Teheran).

In the example scenario, Red launches a theater nuclear attack on cue from a time tag "on D + 3 at 1200." Red also sends messages to NATO and Blue with an enabling condition "2 days after deployment of forces in the Transcaucasus." At each point the Red Agent scheduler steps time forward, the ACL planners check these conditions.

Each slot has <if>, <then>, <backup>, and <default> elements. The <if> conditions are the antecedents that must be satisfied for a possible slot entry. The script specifies these in the form of conjunctions of conditions, much like the essential conditions for script and branch choice. The <then> elements are the slot entries invoked if the <if> conditions are satisfied. The <backup> elements are alternative slot entries to try if the <then> element does not result in a satisfactory look-ahead outcome. The <default> element is the slot entry to be used if there is insufficient processing time to test the conditions, or if the situational conditions match none of the <if> clauses. The elements of an example action are shown in Table 1.

Once all the slots of an action are filled out, the ACL passes the actions to the TCL for further refinement and projection. The TCL applies rules concerning local tactical doctrine and sends the completed action, the current Red situation estimate, the period of projection, and the local goals to the Force Agent. Force projects the situation up to the next action (either Red or Blue), returning the updated situation to the TCL. The Force estimate itself may be uncertain, taking the form of a range or distribution.

Branch, Continuation, and Move Tests

Branch, continuation, and move tests provide the primary control structure for moving through the scripts. All of these tests involve checking to determine if the situational conditions match templates associated with particular actions (see Selection Criteria and Appendix C for a description of this process).

The branching criteria tell which track to follow in a script or when to switch to an entirely new script. Branches differ from actions be-

Table 1

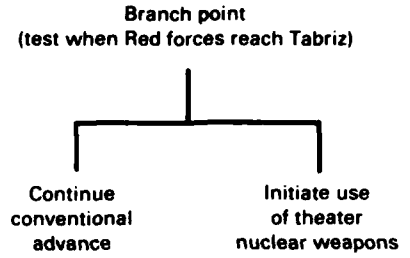
ELEMENTS OF EXAMPLE ACTION

Element	Action
TIME	At D - 30
IF	If roads are open in the Transcaucasus and Iranian forces are below two divisions in Khorasan
THEN	Deploy three MRDs[a]
BACKUP	or else deploy five MRDs
DEFAULT	Deploy three MRDs

[a]Motorized rifle division.

cause several entirely different sets of options may be followed from the branch point on. An example of a branch test is shown in Fig. 3. The ACL follows the script, filling out actions and updating the world model until the branch point is reached. At that time, the ACL chooses the branch whose antecedents best match the situational conditions. Some criteria, such as whether Blue reinforcements will arrive, may be determined from queries to Force. Other criteria may be indeterminate and must be assumed. Then the ACL marks the branch with a tag for later testing of the assumption. The NCL performs the branch tests if any of the branch options involves strategic actions or jumping to a different script.

Continuation tests are somewhat like branch tests, except that they break out of the context of the script completely. Continuation tests check to see if the current script is still appropriate. The types of criteria for continuation include sufficient resources, proper time window, and correct situational conditions. If the continuation criteria are not satisfied, the ACL notifies the NCL, which then compares the other possible scripts to see if a more satisfactory one can be found. This process is much more time-consuming than a branch test and so should be avoided if the specific actions can be enumerated. The example includes a continuation test at D − 13: If there is a NATO



Essential criteria for conventional advance option:

1. Ratio of Red's ADEs to Blue's ADEs 1.5:1.

2. Blue reinforcements will not arrive for 24 hours.

Fig. 3—Example of a script branch point

commitment to the conflict, terminate the B script and give control back to the NCL.

Move tests are quite different from branch or continuation tests, as they specify when to give control to the Blue agent. Each time the world model is updated from an action, the NCL checks to see if the new situational conditions match an action condition for the Blue Agent (actually the Red Agent's model of the Blue Agent, because Red is still in look-ahead). If the Blue Agent is known to be following a script, the NCL tests only those action criteria in that script. If the NCL cannot definitely assume a script, it will have to check action conditions for all likely Blue scripts.

Evaluation Tests

Evaluation tests are the most complex of the various tests. The ACL makes a check after each action to determine if an evaluation test is warranted. If so, the test made is a comparison of the simulated future achieved in the world model and the time-tagged goals that the agent must accomplish. The time-tagged goals represent the minimum allowable performance. The goals consist of conditions that must be present at the specified time, such as forces in place, vessels de-

stroyed, or basing rights obtained. If the actions satisfy the goals, no changes are made and the planning continues from that point. If the goals are not adequately satisfied, the ACL consults a table that maps specific unfulfilled goals to responsible actions. Alternatively, the script may have direct pointers to actions that must be modified. In either case, the ACL modifies the actions to their backup values and again simulates the results and checks them against the goals. This process continues until the ACL finds a satisfactory action sequence or until the ACL exhausts the actions and notifies the NCL that it must select a new script.

The test to see if an evaluation is warranted is similar to the quiescence tests made in chess play. A test should be made if (1) a change has been made to the world model, (2) one or more time-tagged goals are present within the evaluation interval, and (3) an exchange of moves has been completed. An evaluation check might be made in our Persian Gulf scenario just before the invasion.

The test for accomplishment of goals is similar to the goodness-of-fit tests described earlier for essential conditions. The goals may be satisfied in several ways through use of OR and AND conditions. For example, the initial strike in the nuclear scenario may have as its goals knocking out 50 percent of Blue armored capability or disrupting 80 percent of C3 capability and 20 percent of armored capability. Satisfaction of either goal results in the action sequence being provisionally accepted.

Alteration of actions is the most involved operation. As mentioned earlier, the script includes a table or mapping that relates specific goal failures to particular actions. For example, we may have a goal of having five ADEs in place by D − 1, but only three are deployed by this time. This goal difference may be related to the mobilization action at D − 30. Assume this action has a backup entry of earlier mobilization, which the ACL now activates and runs through the same simulation and evaluation process. If this new action is successful, the ACL provisionally enters this action and its outcomes into the world model and continues the planning from this point. If the system is unsuccessful with all backup actions, the ACL alerts the NCL. The NCL tags the script as failed and either invokes rules that change the resource allocations or attempts to fit a different script.

Look-Ahead Termination Tests

The Red Agent terminates the analysis for a script when (1) the goals are all accomplished in look-ahead, showing the script to be satisfactory; (2) the evaluation shows the script to be definitely inferi-

or to any other previously evaluated script; (3) the evaluation shows the script to be superior to all other scripts for this situation; or (4) additional look-ahead does not reduce the evaluation uncertainty. Alternatively, the NCL may have some arbitrary depth bound for search, at which point the analysis ends and the program compares the options.

Normally, we expect the Red Agent to investigate several scripts at the beginning of a campaign, analyzing each of them over several moves using look-ahead. The depth of analysis would be governed by the criteria described above. Probably none of the scripts would satisfy all of the Red Agent's goals for the situation, and so the option with the highest degree of goal satisfaction should be chosen.

Action Execution

The above functions all operate in look-ahead, with no actual game time advance. The Red Agent uses the functions to investigate one or more scripts over several decision cycles and selects one script for execution. That script now has all its actions specified for several moves, including such aspects as force levels, timing, and engagement locations.

The NCL now passes the detailed Red action sequence, expected outcomes, time-tagged goals, and expected Blue moves to Force for execution. Force executes the action and monitors for critical assumptions, such as the following:

- The time stream of projected outcomes matches the actual outcomes.
- The time-tagged goals are accomplished.
- Blue makes only anticipated moves.

If any of the assumptions are violated, Force halts the execution and notifies the Red Agent, who then reexamines its options under the newly encountered options.

RED MODEL OF BLUE RESPONSE

During the look-ahead process, the Red NCL must consider the different options open to Blue at each point in order to assess the robustness of the Red plan being formulated. The extent to which Red is concerned about Blue's plans depends on Red's character or decision-making style and on the model that Red has of Blue's character. These agent profiles determine to some extent the amount of processing that must be done at each decision point.

The Red Agent's ideas about Blue behavior are determined by the supervisory analyst, who sets up initial definitions of Red's character to correspond with the purposes of the experimental run. One important variable is the extent to which Red is willing to accept risks: A risk-taking Red Agent will be less interested in Blue's (perceived) low-probability options, and a risk-averse Red will be likely to consider all Blue options, balancing the perceived probability of each option against the final outcome. Another variable is Red's assessment of Blue's will: If the analyst decides that Red should assign a low credibility to such strong actions as Blue intervention, the Red Agent will be less likely to explore those branches in the look-ahead, or will examine them in less depth.

Red's model of Blue's behavior takes the form of scripts that are identical to Red's in format but different in content. The Blue scripts would include plans for defending the areas for which Red attack plans have been prepared, as well as for potential escalation to different theaters or attack axes. For the initial system, we will assume that the Red Agent has access to all of Blue's scripts but does not know which one Blue will select in any given situation.

The Red Agent initially selects Blue scripts based on his assessment of Blue's character and "will." For example, a risk-taking Red who believes Blue's will to intervene is low might select for initial consideration a Blue script that called for diplomatic intervention only, and a risk-averse Red might consider first a Blue script that called for active deployment to the affected area. As the look-ahead analysis proceeds, Red might find that a particular Blue script yields such a bad result for Blue that even the most pusillanimous Blue would reject it; then Red would back up the search to the first point where this script was invoked and select a new candidate Blue script. Slots in the Blue script are filled in as the look-ahead progresses, just as the Red script is fleshed out.

As the exercise proceeds and the Red script position changes from hypothetical to "real," the Red Agent takes its move and observes the actual Blue move. If Blue actions do not conform to the assumed Blue script, the Red Agent must either change parameters in the current Blue model (e.g., keep the same script but change the number of divisions being used), or look through the other available scripts to find one with a better fit, then recompute all subsequent projections based on the new candidate Blue script and the new estimate of Blue character.

Red's personality determines to a great extent the amount of processing that must be done. The simplest Red Agent to run would be a risk-taking Red with a low assessment of Blue's will to resist. This

Red would select only one or a few high-probability Blue options to consider during each phase of the look-ahead. An option perceived as unlikely would be ignored, even if it led to undesirable escalation. At the other extreme, a risk-averse Red with a high assessment of Blue's will would carry along at each branch all the possible Blue options, investigating each to a point where it could estimate the combination of the likelihood of the branch with the desirability of the outcome.

There may be difficulties in deciding how far to take opponent modeling. As an example, if Red is attempting a feint, he needs to be able to guess whether Blue will fall for it. This means Red must explicitly model Blue's model of Red or the effect of this must be written into the script. We expect that the later versions of the Mark III system will treat the explicit type of multilayered analysis. The initial systems will assume a simple Blue response to the situation.

CONTROL STRUCTURE

The Red Agent must coordinate the operation of many different functions—script selection, resource allocation, slot refinement, move tests, branch tests, continuation tests, evaluation processes, and Blue modeling. It can do so in a strictly sequential manner, or it can interweave processes much like an operating system. Strict sequencing is the simplest and most direct. This is efficient if there is usually a natural successor to each process, such as evaluation tests typically following move tests. For the most part, this appears to be true in the initial Mark III system. We plan to use a scheduler that cycles through the processes in a preset sequence, invoking those processes that satisfy activation conditions.

Interweaving of functions by an operating system type of scheduler is a more flexible but more difficult means of allocating control. At any given time, several modules may be competing for attention. Each module may send a request for time and reasons for priority. The scheduler then examines the requests, prioritizes the modules, and allocates time resources accordingly. If in the middle of a process a higher priority request is received, the scheduler may suspend or kill the currently active process and initiate the new one. Such knowledge-driven scheduling is necessary if, for example, new data are received in the middle of an evaluation. Normally, though, all new data are received and all actions taken at the beginning or end of a move step. In later versions of the Red Agent we may include capabilities for intelligence gathering and action execution during own and oppo-

nent moves, making it necessary to have data-driven behavior in place of strict scheduling.

DEGREE OF AUTOMATION

The Mark III Red Agent is designed to be ultimately fully automated. Each step in the script selection and application process can be performed by computer. Nevertheless, there are many places where for ease of development, for flexibility, for training, or for added power the supervisory analyst or other human expert will be able to modify the system's choices or change parameters. Below, we recap the Red Agent's time sequence of steps, describing how each may be performed by computer or by human input. The process begins with the specification of the Ivan and the input of the initial conditions (force levels, political status of the participants, etc.). The supervisory analyst must input these manually at the beginning of a run.

The first automated operation is the Red Agent's selection of a script to pursue. This involves a determination of the candidate scripts, an initial comparison of those scripts, and selection of one for prosecution. Determination and initial comparison of the candidate scripts involves checking the essential criteria to prune down the candidates and then ranking the remaining ones using the optional criteria. This comparison is based only on the currently observable situational conditions and any short-term projections provided by Force. The supervisory analyst may override the estimates and input different information or may simply input a different set of evaluations for the options. Also, some of the selection criteria may not be observable or derivable at this point. The NCL will assume these to be provisionally true and tag them for later checking. The human analyst, upon observing these deferred criteria, may override the program and simply judge the criteria to be true or false (thereby specifying an assumption behind the ensuing analysis).

Invocation of a script results in a specification of the theaters involved, the time sequence of goals for each theater, and the skeletal tree of branches, actions, and slots. The analyst may wish to modify any of these elements, querying the database according to time in look-ahead, specific action type (amphibious attack, air interdiction), or geographic area.

The next step is allocation of resources to the various theaters. The NCL uses special rules to estimate requirements and move forces between theaters, satisfying the requirements of the highest priority theater first. The analyst may view the recommended allocation of forces and modify it manually.

Once script selection and allocation of forces are complete, the ACL begins to fill out the script. This involves performing a number of activities that may occur in almost any order: slot specifications, branch tests, continuation tests, move tests, evaluation checks, action modifications, and look-ahead termination tests. As the program moves forward in time, the ACL tests the activating conditions for each of the processes, invoking them on a first-come, first-served basis.

Slot specification occurs if there is an action that has the current (simulation) time as its activation time, or if the current situation satisfies its enabling conditions. The action may take place in any of the theaters, as the Red Agent processes all the theaters in parallel. The ACL fills in each of the slots of the action, using the condition-action rules associated with the slot. The analyst may override the recommendation and choose some other slot entry. If he does so at the time of the action, no special problems result. If he modifies a slot entry at a later time (after the look-ahead moves beyond the action), the script actions have to be backed up to that point and rerun.

Once the ACL or the analyst fully specifies an action, it is sent to the TCL for simulation by the Force Agent. The resulting outcome prediction is then input to the Red database. The analyst may modify this prediction if he feels it is inaccurate or if he wants to test the sensitivity of the action.

If a branch point is encountered, the NCL or ACL compare the options using the essential and optional criteria, and choose the option with the highest evaluation. The analyst may either change the essential and optional criteria or override the choice. Again, the ACL or NCL may not be able to evaluate some of the criteria at the time of the action and thus defer them for later testing. At the time of the branch, the analyst may preempt the system and judge these deferred criteria to be true or false. Continuation tests result in similar manual inputs as the branch tests.

The NCL automatically performs a move test whenever a change has occurred in the database (as a result of an action or message). The NCL checks the antecedents for rules used by the opposing agent and if one is satisfied, gives control to its model of the agent. The analyst may override this process and either specify a new action or omit a previously planned one. In the early stages of development of the Mark III system the analyst may also tag the move points in the script.

The Blue move itself (as modeled by Red) is made by calling the Blue script with the current situation. Slot entries, branch tests, and move tests are made just as for Red scripts, and the resulting actions are passed to Force. The analyst may instead wish to free-play Blue, modify the script's choices, or change the criteria for action selection.

Evaluation tests and the ensuing action modifications are probably the most frequent and necessary points for expert intervention. The program automatically performs an evaluation test whenever an action is performed, a change is made to the the database, and time-tagged goals are present for evaluation. During script preparation, the analyst may circumvent this process by simply tagging the necessary points for evaluation. When the program encounters an evaluation point (whether automatically determined or manually input), it checks the essential and optional conditions for satisfaction of goals. Here the analyst may change the set of goals or override the evaluation. In either case, if the NCL determines that a goal discrepancy is present, the program attempts to change the actions leading to this outcome by consulting a table that maps failed goals to previous actions. The program attempts the action sequence again and checks the outcome. The analyst may have a better idea of what action the Red Agent should modify to achieve the desired outcome. The analyst thus directly changes the type, timing, or form of force application that led to the failure, and the program backtracks to that point and reruns the analysis.

The final type of test the Red Agent automatically performs is checking to see if the look-ahead is complete. The NCL terminates look-ahead if all goals have been achieved, if the provisional evaluation of the script is definitely superior or inferior to the other script choices, or if excessive uncertainty has developed in the analysis, making further simulation useless. The analyst may designate that the look-ahead be terminated earlier or continued beyond the recommended point.

The various options for automated control and manual intervention are summarized briefly in Table 2.

In most of the above discussions we assume the analyst performs the manual input at the same time as the automated program would perform the operation. In fact, the analyst may change Ivans, change goals, insert actions, or alter initial conditions at any time. If the analyst does any of these, the analysis will have to be backdated to the point of change and the simulation rerun from that point.

Many of the same procedures should be useful for initial specification of the Red Agent rules by the experts. This knowledge engineering function involves initial loading or refinement of the rules and control system parameters in the Red Agent database. As it moves through a script, the Red Agent may query the expert for rules a data, check for completeness and consistency, and display implications of the rules being input.

In the final operational system, frequent analyst intervention will greatly slow the response time of the system. We expect most auto-

Table 2

Automated and Manual Intervention Options

Activity	Automated	Manual Intervention
Input of initial conditions	Not performed	Input by analyst
Allocating goals to theaters	According to table associated with script	Modified by analyst
Allocating resources to theaters	Uses special rules to move forces	Analyst modifies rules or actual allocations
Slot entry	Uses condition-action rules	Analyst overrides choices
Branch test	Compares options using essential, optional conditions	Modifies criteria or specifies actual branch choice
Continuation test	Checks essential conditions	Overrides choice or tags point for test
Move test	Checks opponent antecedent conditions	Overrides choice or tags point for move
Blue response	Fits Blue script and determines actions	Free-plays Blue or modifies automated choices
Evaluation test	Checks for evaluation criteria	Overrides choice or tags point for evaluation
Action modification	Uses table of mappings	Specifies action change directly
Look-ahead completion test	Checks for completion criteria	Overrides choice

mated functions to have cycle times on the order of a few seconds. A single move may involve dozens of look-aheads with repeated calls to Force and Scenario. When the system is used for analysis, the primary human role should be problem definition, specification of the form of analysis, and system monitoring, rather than step-by-step interaction.

IV. EXAMPLE APPLICATION: THEATER NUCLEAR ATTACK IN WESTERN EUROPE

In this section, we will walk through a complete Red analysis to illustrate use of the proposed Mark III planning techniques. The example conflict, somewhat more complex than the earlier Persian Gulf scenario, is a Red invasion of Western Europe using conventional forces and then escalating to theater nuclear weapons. Throughout the conflict, both the Red and Blue Agents will be assumed risk-averse. All operations are performed in look-ahead.

The example begins with a scenario favorable to a Red invasion. We assume that ample Red forces are available in the area, NATO forces are at low levels of readiness, and the Blue is involved in low level conflicts in the Mideast and Central America. The main script options open to Red on D − 30 are shown in Fig. 4.

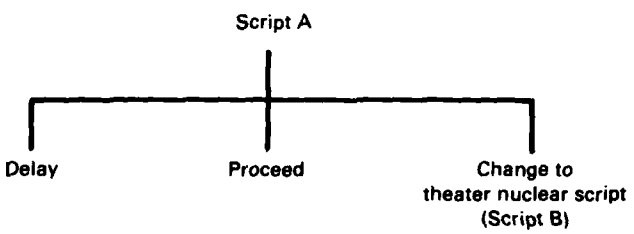


Fig. 4—Red branch options at time D − 1

The scenario marks time in terms of days before or after the main Red attack (D-day). The initial script choice is made 30 days before D-day or at D − 30, because this is the first point (in gaming terms) at which Red must perform an action. The scenario will continue through D + 30, describing several major Red and Blue moves that should result in campaign completion between D + 20 and D + 40. The Red and Blue moves in this analysis are projections by the Red Agent, used to evaluate the script and refine the Red actions.

Each of the four scripts has associated goals. The A and B scripts have the goals of including Western Europe in the Soviet sphere of influence, reducing U.S. military power, and enhancing the superpower status of the Soviet Union. The C script has the added goal of control of SWAsia. The D script has quite different goals: elimination of the United States as a superpower, continuation of the Soviet Union as a superpower in the war's aftermath, and survival of current Soviet leadership.

The C script deserves some special attention. This script could be thought of as two separate parts—an attack on Western Europe and an attack on the Persian Gulf. It could be represented as a combination of two scripts, but there is a large amount of coordination between the two. It appears simpler to embed both campaigns in a single script. In a later, more mature Red Agent, the two campaigns might be treated as individual building-block scripts, designed to be coordinated with a number of other possible scripts.

The Red Agent's first step is to check the essential criteria for each candidate script. These criteria are associated with the script and accessed by the Red NCL. The essential criteria for the A script are a highly favorable conventional force ratio, a high likelihood of destroying most Blue theater nuclear weapons, and a low likelihood of Blue theater or strategic preemption. The B script essential criteria include a favorable theater nuclear force ratio and a low likelihood of strategic preemption. The C script adds the criterion of sufficient excess forces for targeting to SWAsia. The D script, strategic nuclear attack, has as its single criterion the high likelihood of Blue strategic preemption.

Examination of the essential criteria (by checking the initial conditions and by making queries to the Force Agent) indicates that the A and B options are viable candidates. These two scripts also have optional criteria having to do with force levels, political situations, and subjective factors. The NCL weighs these optional criteria and considers the decision flexibility of the choices. The A script is the most flexible of the scripts, as the B, C, and D scripts may still be invoked if the situation worsens during execution of the A script. Such flexibility is measured by the number of possible jumps to other scripts represented in the A script branches. The outcome of this comparative analysis is selection of the A script (conventional attack on Western Europe) with the B script as first alternative.

The A script objective in the Central European Theater is the elimination of all effective Blue military resistance in Western Europe within 60 days from the initiation of attack, with 50 Motorized Rifle Divisions (MRDs) surviving to control the territories. The intermediate objectives in the Central European theater are the following:

- Complete all deployments and have forces in readiness at D − 1.
- Initiate attack 30 days from today's date.
- Effect at least one breakthrough of Blue's main defense line by D + 7.
- Destroy at least 50 percent of Blue's theater-based quick reaction aircraft by D + 10.
- Destroy at least 50 percent of Blue's theater range nuclear missiles (mobile medium range ballistic missiles and ground launched cruise missiles) by D + 10.
- Eliminate all effective military resistance in West Germany by D + 25.

Similar overall and intermediate objectives hold for the North Cape, Thrace, Norwegian Sea, Giuk Gap, North Atlantic, Baltic Sea, and Mediterranean Sea Theaters (see Ref. 23 for a detailed listing of these theater objectives).

The NCL assumes a conventional defense by Blue (this is the best fit to the situation with a risk-averse Sam) and allocates resources accordingly. This process involves prioritizing the theaters by importance, determining the required forces to achieve each theater's objectives, and allocating available resources theater by theater. Determination of required resources involves accessing special rules and interrogating the Force Agent regarding force levels. This results in an initial, rough requirement for each theater (which may be modified during the analysis process). A separate set of rules then moves available forces using such condition-action rules as the following:

> If there is a land-based force more than 200 km inside a theater, let that force be assigned to that theater
>
> If a theater does not have sufficient land-based forces, assign land-based forces from contiguous, lower priority theaters until sufficient

Once the resources are allocated, the NCL and the several ACLs can begin at the initial time (D − 30) and trace out a scenario in look-ahead. Script A calls for a number of actions at D − 30. Among these are the mobilization of forces in the Ukrainian, Belorussian, and Moscow districts for deployment to the Central European Theater and the alerting of Warsaw Pact forces. Each of these actions has slots for refinement. The Central European Command of the ACL, for example, will decide how many troops to mobilize. Filling this slot involves the use of production rules associated directly with the action:

If the NATO forces in Central Europe are below 40 MRD equivalents, let the number of MRDs be 50, otherwise let the number of MRDs be 70

If the RDF is at a high state of readiness, increase the forces by 5 MRDs

It may not be possible to evaluate some of the antecedents for the slot specification rules directly from the Red database. In these cases, the ACL may request estimates from Force. If the proper assignment is still unknown, the ACL may use the slot default value.

Once all the slots for an action are filled, the ACL sends the action, the current Red world model, and the next expected action point to the Force Agent for simulation. Force returns the projected situation (up to the next action point) to the Red world model. The NCL and the various area commands check to see if any unexpected Red or Blue actions may have been enabled during the projected interval. If so, the NCL sets the look-ahead time to that point and performs the action. In the example, no new actions are enabled during the projected interval, so the NCL moves look-ahead time forward to the next action.

As the NCL moves the clock ahead, Red has further moves at D − 25 and at D − 20. These involve major deployments of land, air, and sea forces. Again the area commands fill slots to refine the timing, force levels, and action types.

Between D − 20 and D − 18, Blue should note the Red buildup and alert its own forces. The Red model of Blue (resident in the NCL) has a rule whose conditions match this event, so the NCL declares a move point. We are still in a simulated future projected by Red, and no game time has expired. The Red NCL simply attempts to predict the Blue response, so that it can evaluate the position. For simplicity in the coming discussion, we will call Red's model of Blue simply Blue. At D − 19 in the look-ahead, then, Blue senses the Red buildup and alerts and deploys conventional and nuclear forces in all the theaters. Both the Red and Blue deployments continue until just before D-day.

At D − 1 Red has a branch point. The Red deployments must be complete at this time for the invasion to succeed. Also, the Blue responses must be checked to see if theater nuclear weapons are necessary. The Red branch options are to delay, to proceed with the conventional attack, or to change to a theater nuclear script. (See Fig. 4.) The NCL (rather than the ACL) performs this branch choice as the choice involves multiple theaters and includes a possible switch to a new script. The first option, delaying the attack by one or more days, is chosen if deployments in the Central European, North Cape, or Thrace theaters are incomplete. This delay option has actions of

delaying or altering the attack plans depending on the nature of the deployment failure. This is very similar to an evaluation and action modification (such as deploying earlier). The NCL selects the second, proceed option if all deployments are complete, Blue defenses are moderate, and Blue shows no evidence of a preemptive theater nuclear strike. The third choice, a jump to the theater nuclear script, is made only if Blue appears ready to preempt or if Blue deployments are quite strong.

Choice of the theater nuclear script at this time is different from initial choice of that script. The force deployments and cues to Blue are different, presumably with a lower likelihood of escalation. For comparison, the NCL may later investigate a D − 30 initial application of the theater nuclear script.

Assume that the look-ahead at this point does indicate a high likelihood of Blue theater nuclear preemption. The Red NCL jumps to the theater nuclear script, with its new goal of control of Western Europe within 30 days. Instead of a purely conventional attack on D-day, this script adds the launch of theater nuclear missiles, reconnaissance satellites, and antisatellite missiles. Several ACL branches also come up here, such as launching against land assets only vs. launching against both land and sea assets. The Western Europe area command selects the first option if Blue carrier nuclear assets cannot damage Red nuclear capabilities or if Red theater nuclear weapons are expected to be ineffective against the carrier assets. This evaluation will require queries to Force. This branch, incidently, is not considered as a choice between separate scripts because the same goals are present with either option. The branch options or "tracks" simply represent different means of attaining those goals.

At D-day plus minutes or hours, Blue has the options of continuing its conventional defense, using theater nuclear weapons, and escalating to strategic nuclear warfare. Having observed the Red theater nuclear attack, Blue (again, Red's model of Blue) must select one of these scripts to follow. If the Red attack has succeeded in destroying most of Blue's theater nuclear assets, but the conventional forces are still strong, the first option may be favored (probably coupled with seeking a cease-fire or settlement). If the Blue theater nuclear forces are still operational after the attack and the force ratio is advantageous, Blue may select the second option. Finally, if Western Europe appears lost and U.S. theater forces have experienced major losses, strategic nuclear warfare may be called for. For the purpose of this example, assume that Blue selects the second choice, use of theater nuclear weapons.

At this or a later time, Red has a similar branch point activated by two possible conditions: (1) Blue launches a theater nuclear attack on

the Red homeland, or (2) Blue initiates a strategic nuclear war. In either case, Red jumps to the strategic nuclear warfare script and this again changes the actions and goals for Red. Assume in the example that neither condition occurs and Blue responds with a theater nuclear attack. This and several of the other script actions may overlap in time or preempt each other. Instead of strict initiation times, these actions have enabling circumstances. The NCL and ACL must frequently monitor the changing conditions.

At D + 3, Red has an evaluation point. Red has several intermediate time-tagged goals to test. For example, in the Central European theater the following goals are present:

- Blue's surviving nuclear Quick Reaction Aircraft is less than 25 percent of its original figure.
- Blue's surviving intermediate and long-range ballistic missiles are less than 25 percent of their original figure.

If either goal is not achieved in the look-ahead, Red initiates specific actions—such as revert to D-day and reallocate nuclear attacks. These actions are associated directly with the unachieved goals (see Ref. 23 for goals and action revisions in other theaters for this example scenario).

The Red Agent may stop the look-ahead at this point or continue to D + 30. The NCL should terminate the look-ahead if it cannot satisfy the intermediate goals with any combination of script actions or if too much uncertainty has developed in the analysis. Alternatively, if the NCL continues the look-ahead to D + 30 and determines that the goals are met (to a high degree of certainty), then the analysis can be terminated and no other scripts investigated. At this point, the Red Agent may invoke the actual Red move up to the first move point (by passing the actions to Force), observe the outcome, and continue with the script. If the Blue Agent performs an unexpected action during this actual game operation, the Red Agent may have to revise its estimate of the Blue Agent character. This is done by attempting to fit the observed actions to other Blue scripts or by invoking rules that match the action directly to a specific Sam type. Following such a change, the Red Agent will have to reevaluate the applicability of the current Red script.

V. INTERFACE REQUIREMENTS WITH OTHER AGENTS IN RSAC

The Red Agent is, of course, intimately connected with the operation of all the other agents in the RSAC—Systems Monitor, Force Operations, Scenario, and Blue (see Ref. 2 for a description of the various functions). The Red Agent also requires several interfaces for entry and display of system parameters. Some of the interactions between the different agents are shown in Fig. 5.

The Mark III system differs markedly from the Mark I and II systems in the extent and closeness of interaction of the agents. In particular, the Mark III system requires very high interagent bandwidth

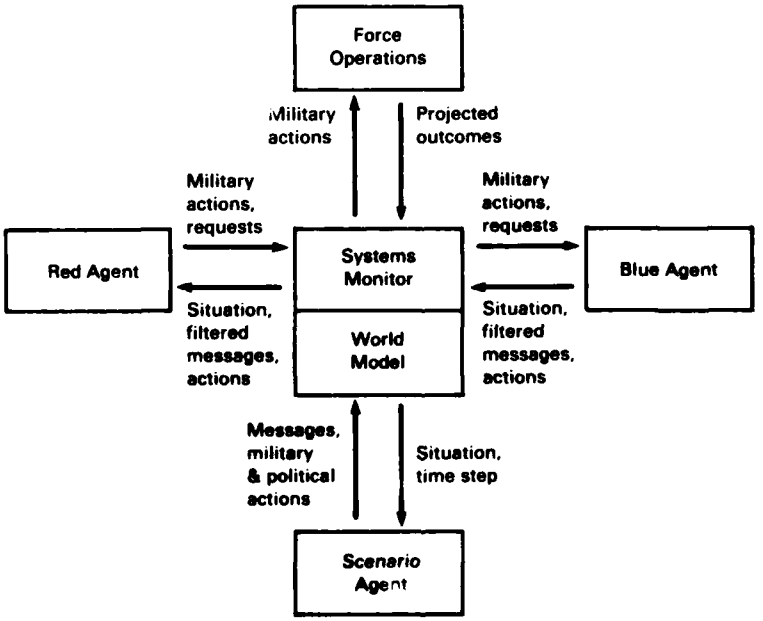


Fig. 5—Interactions among RSAC components

communications because of the large numbers of look-ahead calculations requested from Force and Scenario.

SCENARIO AGENT

The Scenario Agent is the non-superpower model used by the RSAC. It currently consists of sets of rules programmed in Rosie, an English-like computer language. [24] The rules generate actions for each country in response to situational conditions. The actions include changes in country status (e.g., cobelligerent to neutral), sending messages to the major agents, and taking military actions (which are sent to the Force Agent and the outcomes entered into the Red world model). The situational conditions that trigger these actions include Force estimates from the Red world model, messages from the major agents, and time updates from the Systems Monitor.

The first problem we encounter with Scenario is defining the proper level of detail. If we demand the same level of detail as for the major agents, then we need to specify the time, extent, location, and type of all actions by each country. This may necessitate the same type of slot specification as used by the Red Agent, a very time-consuming proposition for the many countries. More aggregated actions, such as responses by the set of NATO countries, should be more appropriate. Because only a small number would be present, such aggregated actions should also be easier to match when fitting Red or Blue scripts or refining actions.

Time considerations drive the form of interaction between the Red Agent and the Scenario Agent. The Scenario Agent may receive an update of the Red world model after each simulated action and determine if non-superpower actions are called for. This will result in unacceptable processing delays if the Scenario response takes more than a few seconds. It may be more effective to have a special subset of abstracted rules resident in the Red Agent that simulate the responses of Scenario. If these rules cannot handle the situation, then the Red Agent should call Scenario, invoking the full set of rules. To further increase efficiency, Red may also specify the subset of countries to be considered.

An additional issue is whether to require Scenario to return confidence or probability estimates associated with its responses. This is similar to Red's considering several Blue responses, each with a different probability. We may have to maintain a number of tracks for the non-superpower countries and select a Red action with greatest expected gain across all the possible tracks. As with the Blue actions,

the probability estimates of the different tracks may be related to their goodness-of-fit to the situational conditions.

FORCE OPERATIONS

The Red Agent treats the Force Agent as something of an input-output model. Force has access to all situational conditions and timing data and it has procedures for specifying outcomes of military actions.[3] In the simplest case, the Red Agent inputs a specific military action, and Force simulates the outcome with respect to the current military situation and returns a projected new situation. The traffic for this type of interaction may be very high, on the order of tens to hundreds of Force calculations for each script.

Force may also modify actions to achieve some objective or answer some query. The types of rules the Force model must use are often algebraic in nature (mobilization calculations, force ratios, etc.). These are the types of functions that can be systematically modified until they achieve or maximize some objective. The Red Agent may thus input the actions and short-term objectives to Force and specify which actions may be modified to achieve the objectives.

One of the RSAC design options is determining the balance of action specification by the major agents versus the amount of optimization by Force. In the early development stages we expect that detailed specification by the agents will be considerably easier than programming general optimization rules into the Force Operations Agent. In later versions, it may turn out that a locally optimizing Force model will substantially reduce communications between modules and lower the agent processing requirements.

Force also has the responsibility for execution of actions during an actual game time advance. The NCL passes the fully specified actions, projected outcomes, and expected Blue responses to Force. Force then executes the actions and monitors the results, notifying the Red NCL if an unanticipated event occurs. This execution function should be the least demanding of Force's responsibilities.

SYSTEMS MONITOR

The Systems Monitor updates the game clock, calls for major agent moves, and maintains game records.[4] This is only for "real-time" operations, when the conflict is actually progressing through time. When time is frozen and the Red Agent is in the look-ahead mode, the

agent itself must handle all timing functions internally. The Red Agent must therefore maintain a record of the following activities during look-ahead operations.

- Red and Blue options considered.
- Options selected, discarded, or used as back-up.
- Assumptions held for later checking.
- Goals accomplished.
- Move times.

The Red Agent will store all of these records as tags on the script actions in its database. The Systems Monitor must store similar information about operations during actual game time advances. The Red Agent will subsequently access this information.

HUMAN INTERFACE

The supervisory analyst or the Red Agent operator must be able to designate a point in the script, request a summary of the system status at that point, enter new parameters or rules, and view the implications of those changes. The following are some interactive functions necessary for supporting such operations:

- Capability to halt a run at any time or condition. This requires establishment of menus or interactive graphics for specifying locations within a script, along with means for suspending and resuming processes.
- Capability to change Agent characters, rule entries, or system parameters during a run, and execute specific actions.
- Capability to query the logic behind a rule. The analyst should be able to view the chain of assumptions leading to an action. This requires backchaining of the rules that fired leading to the action, responses from Force and Scenario, and snapshots of the Red World Model (at least those portions relevant to the action).
- Capability to request sensitivity analyses of key strategic and tactical factors. The analyst will often need to observe the projected sensitivity of outcomes to changes in initial conditions, actions, and other variables.

The first two functions are essential to operation of the prototype system. The initial code will be developed with these functions as integral capabilities. The third and fourth functions, explanation capabilities and interactive sensitivity analysis, are somewhat more

difficult, requiring extensive development for useful operation. We expect to incorporate these capabilities in the last phase of our development cycle (see Sec. VI).

Even the initial three functions will require a very facile interface, able to display the current problem solving situation and system status. The types of displays planned include the following:

- Graphic situation display, showing military forces, movements, targets, controlled areas, and engagement outcomes. Separate displays should show the actual situation, the Red-filtered view of it, and the Blue-filtered view.
- Textual situation display, showing conflict characteristics, matching of scripts, branches and slots, current goals, resources in each theater, and operations being performed by the Agent now active.
- Script status display showing in tree form the current position and time, the point in the move cycle, all currently viable options, and the queue of upcoming activities.
- Graphic sensitivity analysis. Plot of outcomes against assumptions. The assumptions may be discrete (e.g., use conventional, chemical, or nuclear weapons) or continuous (number of MRDs in attack). The outcomes may similarly be discrete or continuous.

The first type of display is already present in the Mark I and II systems. The second and third types of display, showing the analysis status, will be similar to that developed in various decision support systems (see, for example, Refs. 13 and 25). An augmentation to the system status display would be the listing of rejected options with the reasons for their rejection. Then if an assumption leading to rejection of an alternative proves to be false at some later point in game time, the operators would be able to resurrect the alternative. The last display type, graphic sensitivity analysis, will be produced late in the development cycle and will be quite specific to the area of strategic planning.

VI. CONCLUSIONS AND PLANNED WORK

The automated Red Agent we have outlined is rather ambitious. It is essentially a goal-driven branched script system capable of close interaction with human experts. The system is designed to perform both information gathering and action selection, represent uncertainties in the environment, predict its opponent's responses, model different decisionmaking styles, and coordinate activities in multiple theaters.

We chose a script-based methodology for this endeavor because scripts efficiently represent complex time-sequenced sets of actions and produce fairly transparent decision behavior. In fact, conventional strategic planning often uses war plans closely resembling scripts. Of course, the Red Agent must do much more than simply follow a canned set of scripts. It must maintain an evolving, uncertain situational model, predict Blue and non-superpower responses to its actions, and revise its plans until its goals are met. In many ways, the Red Agent is building a game tree, specifying, refining, and evaluating its options until one course of action becomes favored. The process of building this decision tree is nondeterministic, and the Red Agent must schedule a number of competing processes, such as time calculations, branch choices, move tests, evaluation processes, action modifications, and opponent modeling. We have outlined some strictly sequential and some knowledge-based procedures for scheduling these processes.

The proposed Mark III system should result in substantially improved performance over the Mark I and II versions, although possibly at some costs in speed and increased system complexity. As shown in Table 3 the proposed system will increase transparency over previous versions through the use of scripts, its three-level design (allowing compartmentalizing of knowledge), and its augmented human interface. Flexibility will be improved by parameterizing many aspects of the scripts, among them the timing and specific characteristics of branch choices, action slots, and evaluation points. Military detail will be handled more directly at the theater and tactical levels and will include modeling of many subjective variables (opponent "will," intentions, etc.). The Mark III system will also directly model the opponent and will systematically alter its actions when projections indicate that interim or final goals will not be achieved. The

Table 3

RED DEVELOPMENT OVERVIEW

Criteria	Mark I	Mark II	MARK III Objectives
Multi-level	No	2 levels	3 levels
Transparency	Poor	Good	Good (with more depth)
Flexibility	Very good	Good	Good (can "tune" plans)
Treatment of asymmetries	Yes	More by rules	More by three level system
Opponent gaming (chess-like)	Minimal	Possible but difficult	Yes
Ease of extracting expert knowledge	Very tough	Variable	Much better
Speed	Fast	Faster	<10 minutes/move
Treatment of military detail	Fair (nuclear) Poor (theater)	Fair (depends on script)	Good
Ability to learn	No	Minimal	More by model of opponent
Sensitivity to subjective variables	Minimal	Fair	Directly addressed
Development of realistic time-phased strategies	No	Yes (script-limited)	Yes (more doctrinal content)

system will also adjust its model of the opponent when its expectations are not borne out. All this is not without cost, of course. We may pay for the increased depth of analysis with speed.[1] The design calls for large numbers of interactions with Force, Scenario, and Systems Monitor.

[1]Because of different design configurations and support requirements, it is not clear at this time whether the Mark III system will be slower than the Mark II system. Improvements in the computing environment indicate that with the same depth of analysis, the Mark III system would be much faster than the Mark II. However, the multiple look-aheads associated with a comprehensive Mark III analysis may result in a slower response time.

We perceive the Red Agent to be an evolving system, acquiring expertise through close interaction with human experts. The initial system will be loaded with scripts for a limited number of conflict types and geographic areas. Its behavior will be highly deterministic, capturing only a portion of the uncertainty present in the environment, the opponent responses, and the intelligence inputs. As the system matures, greater power and fidelity will be achieved through data inputs, rule changes, and control structure evolution. In the next two sections, we describe a proposed baseline system and an evolutionary plan for development.

BASELINE SYSTEM AND DEVELOPMENT PLANS

Our design for the Red Agent begins with development and demonstration of a baseline system (termed the Mark III.1), followed by an evolutionary program for development and testing of the mature Mark III system. The Mark III.1, intended for testing and refinement of our design concepts, will be intermediate in scope between the currently operational Mark II (pattern-based) system and the final Mark III system. For example, it will exhibit the following enhancements:

- Automated allocation and (reallocation) of resources to theaters and fronts.
- Automated allocation (and reallocation) of resources to theaters and fronts.
- Explicit, scripted modeling of Blue's actions.
- Gaming simulation of Red and Blue moves in look-ahead.
- Automated goal checking and backtracking.

The following paragraphs describe these Mark III.1 capabilities in more detail, along with additional functions planned for the later, more complete Mark III system (primarily a more comprehensive form of uncertainty representation, multiple command levels, structurally equivalent Red and Blue Agents, and more extensive facilities for explanation and human intervention.

Identifying Candidate Analytic War Plans

This is the first operation by the Red Agent when confronted with an initiating situation—it must identify AWPs (as defined by a set of primary and anc[illegible]ry scripts, appropriate for investigation. This is done before any look-ahead operation and relies only on information derived from the situational conditions and queries to the Force

Agent. In the Mark III.1 version, this is expected to take the form of a semi-automated process. The Red Agent will attempt to match if-then rules to the circumstances and use the resulting matches to cue the analyst, who will then accept or override the AWP rankings. The later Mark III version will more fully automate this process, using essential and optional conditions to cull unacceptable AWPs and make tradeoffs among the remaining candidates.

There will also be differences between Mark III.1 and Mark III in the modularity of scripts. The Mark III.1 will frequently resort to monolithic scripts that specify actions and branches for a number of theaters and force types (land, naval, air). Use of a few such scripts simplifies matters compared with a more modular approach using many separate scripts, as the scripts internally represent all coordinations between areas and forces. Such large, specialized scripts are difficult to generalize, however. Entire new scripts usually have to be synthesized for any variant on the conflict. The mature Mark III system will rely to a much greater extent on small, modular scripts, which may be linked together to handle various type of conflicts. We expect such building block scripts to require many rules for coordination.

Automated Resource Analysis

In Sec. III, we described how each candidate AWP must have some means of ensuring adequate force levels that will achieve its goals. This entails determining the force ratios required for each area's operations, translating these ratios into required Red force levels on the basis of assumed Blue defensive force levels, checking to see if adequate Red forces are already available within the specific areas, and if they are not, moving forces from neighboring areas. This force allocation analysis may all be done before an action or, as in the planned Mark III.1 implementation, a pre-planned sequence of allocations may be tested. In this simple procedure the Red Agent will use rules to roughly estimate the needed force levels, and then perform a look-ahead with 1/4, 1/2, 3/4, etc. of the needed forces, until the goals are met. This hit-or-miss method should minimize the number of rules needed for estimation and movement of forces. The mature Mark III system will also verify its force allocations through look-ahead but will perform a much more complete allocation analysis beforehand and will use specific rules for reallocating forces in the event of failure.

Using Look-ahead for Analysis

Throughout our discussions, we have emphasized the importance of look-ahead as a technique for testing the potential value of a war plan. The Red Agent postulates an action, examines possible Blue responses, postulates Red actions in response to these, and so on. We have also pointed out that look-ahead can be simple or complex. The Red Agent may look only one move ahead, assume a single Blue response, and evaluate Red progress. On the other extreme, the Red Agent may examine all possible Blue responses, assign probabilities to each, and look several moves down each branch, finally folding the tree back to determine the best action (or the least risky, or probably most successful one).

The Mark III.1 Red Agent will tend toward the former, simpler methodology. The Red Agent will look several moves ahead but will consider only one Blue response to any action, the one that best fits the situation. Red's Blue, in making the best fit, will not perform its own look-ahead. Also, the Red and Blue Agents will act in a lock-step fashion. Each will know all the actions open to the other, and when those actions might be taken.

The mature Mark III Red Agent will be much more flexible. Depending on the Ivan, the Red Agent will examine one or many Blue responses to each of its actions. The Blue Agent will, in a complementary fashion to Red, be free to perform its own look-aheads. Among other things, this will require Systems Monitor to maintain a number of situation databases. There will have to be a database representing the actual situation, a Red-filtered view of the actual database (updated with each projection from Force and Scenario during Red look-ahead), a Blue-filtered view of the Red-filtered database (for Red's model of Blue), a Blue-filtered view of the actual database (for Blue look-aheads), and a Red-filtered view of the Blue-filtered database (for Blue's model of Red). Each of these filtered views should be a "masking" of certain elements of the original situational database.

Move Sequencing

This is an important part of both the look-ahead process and the actual game progress. Control will be given to Red, Blue, and Scenario by "waking up" these agents when it is time for them to take an action. This process should be different for the prototype and final Mark III systems. In the prototype, after the Red, Blue, or Scenario Agent executes an action, it will simply enter into the database the conditions for it to assume control again. These conditions may be

strict times (at 0600 on D − 10) or they may be situational conditions (when force ratio reaches 3:1; when FEBA passes Teheran). Force or Systems Monitor will check these conditions and wake up the appropriate agent when there is a match. The mature Mark III system should expand this to include a wider range of wake-up conditions—when expected conditions are not met, when key conditions are unknown, when a certain time has passed without event, etc. Also the different agents should be able to activate each other directly. For example, If Spain (modeled by the Scenario Agent) sends a message to Blue of its intention to change combat status, it will also send a message to Systems Monitor to wake up the Blue Agent after the appropriate interval.

Learning

The initial Mark III.1 Red Agent will exhibit several rudimentary forms of learning. It will adjust its behavior when its projected outcomes do not match its goals and it will change its assumptions about Blue strategy if Blue behaves in an unexpected manner. Later implementations may incorporate more complex forms of learning, such as automatically altering the normal branch or slot choice after discovery of a successful modification. The system may generate the modification from the automated backtracking procedure described earlier, from analogic reasoning,[26] or from direct expert input. Also, when the NCL determines that its model of Blue is no longer accurate, it may use the information to automatically change the faulty rules used to identify Blue character.

In summary, the initial Mark III.1 Red Agent will be a highly simplified version of the Mark III Red Agent described in this report. It will embody the key automated functions of the Red Agent system design but will not have the potential flexibility, power, and application of the final system. It will also be used solely for development purposes. The later versions will be used for substantive analysis and education.

IMPLEMENTATION PLANS

We expect to have a closely joined developmental and experimental program. The initial implementation of the system will be limited to a specific geographic area, with hard situational information and lock-step opponent responses. This implementation will be used primarily to test the structural robustness of the prototype system—the

ability of the Red Agent to execute a script reliably under several distinct situational conditions. During this initial period, shortcomings of the system will be handled by interactive modification of the rules, parameters, and control structure of the Red Agent.

Structurally, the prototype Mark III.1 Red Agent will be composed of the following elements:

- *A database* maintaining Red's situation estimate, current processing status, models of Blue and other involved countries, outcome projections returned from Force, and analysis history. This database may be entirely present within the Red Agent or portions of it may be accessed from databases maintained by the Systems Monitor.
- *Rules for the NCL, ACL, and TCL functions.* These sets of rules perform the problem solving functions of script selection, resource allocation, script application, and strategy evaluation. The rules will not actually reside in separate modules in the prototype system but will be organized in the form of a main program and subroutines.
- *Interfaces with Scenario, Force, and Systems Monitor.* The interfaces will pass such data as the current Red situation estimate, Red and Red's Blue actions, and events to be monitored. The Red Agent will frequently query the other agents regarding projected actions and outcomes.
- *Interfaces with supervisory human analysts.* These interfaces will allow human intervention at each step of the analysis cycle and will produce limited explanations of the reasoning and implications of each choice.

The initial experimental plans, using the prototype Mark III.1 configuration, will be concerned with varying the situation conditions—force levels, time periods, political conditions, alliance structures, etc. These conditions will be varied but still held within the range of the available scripts. The conditions will be known with certainty by the Red Agent, as will the Blue responses. We will develop these scripts with a specific problem and Ivan and will test them by comparing the resulting actions with actions recommended by experts on Soviet military decisionmaking.

We foresee the next stage (again with the prototype configuration) to involve a wider range of behaviors but still to assume a known Blue response to each Red action. The conflicts will broaden to multiple theaters with the associated coordination problems. Escalation will occur through several weapon levels—conventional, chemical, and nuclear—with deeper look-aheads than in the initial phase. Different Ivans and Sams will be tested, with the assumed Sam changing dur-

ing a conflict. In this phase, the system will still operate on hard information. There will be no uncertainty in the force situation, scenario responses, calculated outcomes, or intelligence inputs.

The third phase should introduce situational and intelligence uncertainty along with completely equivalent but asymmetric Red and Blue Agents. The Scenario and Force outputs will have associated confidence ratings that will be incorporated into the Agent's belief systems. The use of unconstrained, complementary Agents will necessitate development of rules to avoid problems of deadlock and cycling. In this phase, we also plan to develop and test a full range of interactive explanation and sensitivity analysis capabilities.

A full-scale experimental study is possible following the completion of the full human interface and the incorporation of the automated, complementary Blue Agent. The experimental variables should include situational conditions, look-ahead depth, criteria for move points and evaluations, and form of the Blue model. The system performance should be tested against expert performance. The experimental program with the mature Mark III system should also include secondary evaluations of system performance in the areas of sensitivity analysis, force deployment aiding, and strategy analysis.

DESIGN CHALLENGES AND TECHNICAL RISKS

The Mark III Red Agent design proposed in this report is a novel and technologically difficult one, combining approaches from several different areas—scripts, production systems, and goal-directed search. This design assumes near-term answers to many questions, some of them already raised in the previous sections. The most important questions concern the depth of modeling, forms of uncertainty representation, modularity of scripts, and modeling of Red/Blue asymmetries.

The first of these questions, depth of opponent modeling, concerns the number of layers necessary for effective adversarial decisionmaking. We initially plan to implement a Red's model of Blue, but we must also consider the need for a Red's Blue's Red. The problem of representing and analyzing the process of deception argues for the latter, more complex opponent model. The choice will require further analysis and system experience.

The problem of uncertainty representation, discussed in Sec. III and Appendix B, is extremely difficult to solve analytically. In fact, many artificial intelligence systems have required development of "custom" procedures for representing, updating, and manipulating beliefs or

uncertainties.[42] The RSAC developers must decide whether to produce a simple ad hoc system using heuristic rules for updating uncertainty estimates or to develop a more comprehensive and formal system indicating the degree of support and denial of propositions, along with some means of dealing with data contradictions.

The third question, concerning the degree of script modularity, is primarily a problem for the rule writers rather than for those implementing the structural elements of the system. Two difficulties are present—how to coordinate multiple scripts in a campaign and how to move at any time from one script to a completely different one. The first problem is simpler, as such interscript coordination should just entail timing of actions along with checks to make sure outcomes of one script do not violate bounding conditions of another. The timing and checks will have to be written with sufficiently general conditions to allow coordination with many other scripts. The second problem, changing scripts in mid-campaign, is more difficult, as it requires checks on the situation, progress, resources, and other factors. If these conditions do not match those necessary for the new script, the Red Agent must consult rules for adjustment. The rules must be sufficiently flexible to allow entry into the new script at any of a number of points. Again, experience with a variety of script types will be necessary before firm guidelines for script writing can be established.

Questions concerning Red/Blue asymmetries are probably the most complex issues facing the Red Agent and the RSAC. For simplicity, in the report we have emphasized the Red Agent design issues. In fact, the structure described must also model the decisionmaking of the Blue Agent. Strong differences are present between Red and Blue in the form of offensive and defensive postures, different types of goals, amounts of reliance on planning time lines, and degree of centralization of authority. We should be able to accommodate these differences within the Red Agent structure described, but validation will require implementation of unconstrained, complementary Red and Blue Agents, a step late in the planned development and testing cycle.

Finally, the extensive analysis required of the Red and Blue Agents will affect the response time of the system. Even with most of the coding in a fast language, some tradeoff will have to be made between modeling detail and processing time. For example, the early implementations may be able only to model conflicts at the strategic level. Also, the rule writers will have to put a greater emphasis on producing intelligent rules for initially selecting actions (pattern-matching to the situation) than on frequent look-aheads and backtracking. Well-written rules for fitting an action to the situation should be substantially more time-efficient than exhaustive generate-and-test search techniques. Again, resolution of these issues will require experience and testing.

Appendix A

ALTERNATIVE TECHNIQUES FOR RED AGENT AUTOMATION

This appendix compares the applicability of various problem solving techniques for automation of the Red Agent. As mentioned in the introductory section, there are many criteria for selection of an appropriate technique. The system design should be multilevel (able to handle both strategic and tactical operations), transparent, repeatable, rational (act in a manner both logical and consonant with expert strategists), able to model opponent actions, and capable of representing time-sequencing and uncertainty. At the same time, the system must be realizable. It must be able to run in real time and interface with the updated Force and Scenario programs.

We have examined many existing methodologies for problem solving that might be applicable to this problem. Among these are search techniques, production systems, script-based approaches, decision analysis methods, and pattern classification techniques. In the following sections, we will describe each of these techniques, analyze their potential for use, and propose a hybrid technique for implementation in the Red Agent.

SEARCH

Search is the most exhaustive of the problem-solving techniques. Here we define the possible set of states for the situation, operators for moving from state to state, and evaluation functions determining if the goals have been met. The special technique of game search [8, 9] may be necessary because of the presence of an opponent. Game search differs from non-opponent search techniques (means-end analysis—MEA—backtracking, heuristic search, among others) in that it emphasizes producing best responses to an opponent's move, and the quality of a strategy depends only on the terminal result, not on the cost of the path.

The nongaming search technique of means-end analysis is fairly straightforward. Here the difference between the current state and the goal-state is calculated, and the best operator for reducing this difference is chosen.[27] If the best operator is not immediately appli-

cable, its unsatisfied conditions are established as new subgoals. Various heuristics are used to organize the subgoals and reduce search overhead. Carbonell [26] also extends this technique to include reasoning by analogy: A solution sequence that is successful in another similar problem may be transformed to apply to the new problem. In this way, a set of prototypical Red strategies may be defined for key situations and then systematically transformed to operate in the situation in question. As yet, though, this technique has not been applied to any real-world problems.

A second type of nongaming search applicable to Red's decisions is heuristic search. Here task-dependent information is used to minimize (1) the cost of search required to obtain a path to the goal and (2) the costs of the path itself.[8, 9] At the risk of possibly missing a solution, we may accomplish some speedup using special pruning techniques. This technique and MEA normally do not take the opponent's responses into account, so these techniques could result in actions predictable by the opponent.

Game search does take the opponent into account, using such techniques as minimaxing (choosing the option with the lowest Red gains) and Alpha-Beta (pruning those options whose provisional evaluations are dominated by another option). Although these techniques are extremely flexible, they have many problems. Look-ahead may be required over many decision cycles, resulting in a combinatorial explosion and long processing times. The procedure works only if there is a good terminal evaluation function specifying the strength of the position at any point. Also, the program must be able to represent all intermediate states on the path to the goal.

There are additional problems with both the game search and nongaming methodologies. Each situation and action must be explicitly represented. Actions made by different groups must be coordinated in some fashion. Time is not normally represented, as an action simply takes place when its triggering condition is present.[12] Finally, the process is not particularly transparent. It is difficult to see how a consistent strategy emerges from the aggregation of many separate goal-directed actions by each theater or functional group.

PRODUCTION SYSTEMS

Knowledge engineering systems are often based on the organized use of condition-action or production rules. Each distinct situational condition is associated with an action sequence.[27] These rules encode knowledge about situational conditions, about how the situations

change, or about the effects of goal-oriented actions. Information regarding sequencing and evaluation are often implicit in the rules.

The advantage of condition-action rules is that they can encapsulate deep ideas efficiently and transparently.[28] A set of rules can guide behavior without the need for representing intermediate states or specific goals. A production system simply consists of a collection of condition-action rules, a work space representing the environment, and a control mechanism.[11] The rules can even be incomplete and still function adequately.[29]

The organization of rules in knowledge sources can be very flexible. The rules can have associated confidence ratings, such as those used in MYCIN.[16] The rules may be organized hierarchically, as in HEARSAY-II.[18] Here knowledge sources examine entities at one level and hypothesize or confirm those at another level. The rules may be used in a forward-chaining fashion from assumptions, in a backward-chaining fashion from goals, or by middle-chaining from "islands" of high confidence.[31] The rules can even have metarules that look at the antecedents of the normal rules, determine if the antecedents are too difficult to evaluate, and distribute processing resources accordingly.[32]

In our application, the rules might take the following stylized form shown in Table A.1. Very complex rules involving conjunctions of conditions may be represented in the same way. For example, in the first rule, the tanks may be massing or attacking.

Certainty values can be associated with each rule and a calculus used to propagate these certainties. This facilitates development of a probabilistic model of the situation by the Red Agent. Such a model can be very useful for deciding whether to gather information or initiate actions.

Table A.1

POSSIBLE PRODUCTION RULES FOR TACTICAL PLANNING

<object>	<attribute>	<value>	_____	<action>
tanks	massing	region 7	_____	Initate air attack region 7
troops	airlifted	unknown destination	_____	request intelligence foray
messages	transmitted	blue access	_____	assume basing request

The sequence of activities in such a production system is then as follows:

- Access the current goal tree for the Red Agent.
- Retrieve the list of rules whose conclusions bear on the current goal or whose antecedents match the current conditions.
- Order the testing of the rules using metarules.
- Evaluate the rules.
- Set up subgoals if there is not enough information to evaluate a rule.
- Continue until a course of action is selected that satisfies most goals.

This procedure has the advantage of facilitating question answering because of its standardized structure. However, time may be difficult to represent, individual actions may be hard to coordinate, the complete situation vector must be pattern matched at each move, and not all conditions will fit easily into the above format. Also, the incorporation of new rules may be difficult because of the typically amorphous structure. Finally, game strategies such as minimax are not easily incorporated into the rules.

SCRIPTS

A script-based system is similar to a production system, with situation matches triggering actions.[7] The differences are that (1) the actions are much larger, connected sequences of operations with explicit time tags; (2) the control mechanism of the production system is replaced by branching points in the scripts, invoking different "tracks" in the scripts or different scripts entirely; (3) the actions specified in a script are refined by filling in slots after the script is selected; and (4) the script defines a context, so that situation pattern matches in the middle of a script need be made to only a few diagnostic features.

A good example of a script-based system is POLITICS.[6] Each major strategy (decoy, fall-back defense, invade, *etc.*) is represented as a script. The slots in the script denote the specifics of the strategy. Both the script choice and the slot specification are derived through use of condition-action rules. The rules may specify actions to perform intelligence gathering, to infer the opponent's strategy, or to perform actions.

The rules for action invocation are composed of four parts:

1. Trigger: These are rapid, inexpensive tests to check if the rule is applicable. The tests are organized as a discrimination network (sequences of yes/no tests).
2. If: These are more complex antecedents tested only if the trigger conditions are satisfied. If the If conditions cannot be evaluated, they are set up as subgoals.
3. Then: This is the general action set, specifying which actions to perform.
4. Refinement: Here additional and more specific condition-action rules are used to fill the slots in the script.

All of the other advantages of production rules—transparency, modularity, toleration of inconsistency, generality to new conditions, etc.—are present with scripts. Additional advantages are producing a coordinated, consistent response in complex planning situations, having the control structure implicit in the representation, and explicitly representing time sequencing of actions. The disadvantages are the lack of gaming in the action selection, the requirements for eliciting very detailed scripts for each distinct area or function in the conflict, and the somewhat inflexible response a script produces for a given situation.

DECISION ANALYSIS

Decision analysis is somewhat different from the above techniques. The intent is to represent all major actions and their probabilistic outcomes in the form of a decision tree, and then to select the action at each choice point with the highest expected gain. Alternate layers of the tree indicate own and opponent moves and outcomes. Assuming that the decisionmaker can elucidate all possible actions and counteractions, can estimate probabilities and utilities of all outcomes, and is rational (the decisionmaker will always choose the action with the highest expected utility), we can roll back the endpoint values and pick the best action.

Automated decision analysis programs have been applied to tactical problems,[14, 33, and 34] and to international crisis problems.[13, 25] Pearl [9] demonstrated that we can augment the traditional decision tree with an and/or goal-directed representation, resulting in a marriage of MEA and decision analysis. His system, termed a goal-directed decision support system, is used primarily for automated elicitation of decision "frames." Other specialized techniques use probability

ranges instead of point estimates [35] and probability influence diagrams in place of decision trees.[13]

The emphasis in decision analysis is the choice of the best course of action after all alternatives and outcomes have been explicitly represented. We start by defining the tree of actions, responses by the other agents, and outcomes. We then must specify the conditional probabilities of occurrence of the outcomes and their values (according to some goal function). When this is all established in tree form, we can:

- Integrate new information by updating the outcome probabilities according to Bayesian or other methods.
- Choose the best intelligence gathering option by performing a value-of-information calculation.
- Choose the best action option by rolling back the tip node evaluations to the current root node (as the scenario progresses, we move through the tree).

If the actions open to the Red Agent satisfy the requirements of additivity, utility independence, etc. demanded by decision analysis, this can result in a complete and well-supported analysis. Unfortunately, most problems do not lend themselves to this format. Also, the approach requires large amounts of subjective input, which may require revision as the scenario progresses. Many of the other techniques described are much more parsimonious in their processing, as they require only the most promising portions of the option trees to be elucidated at any given time. Finally, decision analysis is primarily a mechanism for comparing alternatives, where here we frequently need a generator of alternatives. In the RSAC system, the emphasis is on techniques for generating plans that are robust, efficient, and reliable rather than on means for comparing fully specified plans that differ along those characteristics.

PATTERN CLASSIFICATION

The Mark I and II Red Agents did a nearest neighbor match to a set of prototypical situations (approximately 400 in all). This approach had several shortcomings. It seems unlikely, for example, that the same set of dimensional weights should be appropriate to all situations. It would probably be better first to define the major class of situation—attack, retreat, resupply, etc.—and then attempt to use a specific distance metric within that class. In this way the distance weights can depend on the general situation. Some distances may not even make sense in a given situation and should be deleted from the vector.

Even so, the simple selection of a course of action based on a pattern is limited to situations in which a single response is appropriate. If coordination of actions over time is necessary, a script-based approach would be called for. If specific objectives must be met, a simulation look-ahead with an opponent model may be necessary. Finally, to specify an action for each possible distinct situation, an unmanageably large number of actions and patterns would be needed.

CHOICE OF A TECHNIQUE

The demands of Red Agent planning quickly narrow down the set of useful techniques. The complexity and time coordination requirements make intractable the fine-grained approaches such as game search, production systems, and pattern classification. Similarly, the rigorous decision analysis approach requires far too much structure and elicitation of parameters (values, probabilities, attributes, and importance weights). The only technique that appears to be capable of representing the complex, time-sequenced actions of the Red Agent is the use of scripts. Scripts efficiently and transparently show the progression of actions and branching under changing conditions that characterize the actions of a major power. Scripts maintain a context, so that situation matching is much less cumbersome than in some of the other methods. Still, scripts must be augmented by simulation-based look-ahead to ensure that the Red Agent's goals are met by the plan. If not, the system must use production rules to modify slots in the script or to choose an entirely new script. This use of a hybrid system using both scripts and simulation appears necessary because of the opponent situation.

Researchers have dealt with some analogous problems by mixing production systems with other problem solving techniques. PROSPECTOR [36] has used production rules and semantic network representations, and HEARSAY-II uses stimulus-response frames instead of simple triggering conditions for its rules.[18] Similarly, production rules have been combined with simulation. A simulation-based look-ahead program can determine the future situation resulting from an action, evaluating the potential gains or losses. Wesson [12] successfully used this approach to explore the set of possible actions in air traffic control.

We can also augment our proposed combination of branched scripts and simulation-based look-ahead with use of an object-oriented simulation language. Here we view each element in the Red Agent (air-

craft, radar, ground forces, etc.) as an actor [37] or object, as in SMALLTALK,[38] DIRECTOR,[39] or ROSS.[40] ROSS, for example, uses frames to define an object and represents dependencies between objects using constraint propagation. The objects can pass messages, perform decisionmaking (using condition-action rules), and execute actions. The ROSS simulation, currently of air penetration, operates at several levels of abstraction and produces easy-to-understand graphic output.[40] Distributing the problem-solving capabilities across theaters and units in this way should provide a more realistic format than a centralized organization, although at the expense of increased complexity.

Appendix B

UNCERTAINTY REPRESENTATION AND PROPAGATION

Uncertainty is present at all stages in the execution of warfare and its effects must be represented in our script-based models. The principal sources of uncertainty in the Red Agent are the following:

- *Initial situation estimate.* The set of conditions describing the situation at the start of the exercise are a priori estimates. The estimates of force levels, enemy location and movement times are all estimated with some uncertainty.
- *Messages/intelligence about the situation.* New data come in about the enemy situation and are used to revise the situation estimate. The information may be inaccurate, outdated, or nondiagnostic.
- *Quality of fit of Blue responses.* The predicted Blue action may only poorly fit the situation present. We may estimate the likelihood of each Blue response using its goodness-of-fit to the situational conditions.
- *Model calculation uncertainty.* When queried about a specific situation, Force or Scenario may return several possible results, each with a different probability.

Some of the other sources of uncertainty will be ignored, such as the quality of fit of the Red script or the possibility of execution errors. The Red script uncertainty is unimportant because the Red Agent must at some point choose one script to follow. Although execution errors are easy to model, they do not add to the understanding of the strategic planning problem; they simply increase the randomness that must be modeled.

All of the items of uncertainty may be expressed in terms of point probability estimates (e.g., probability of success = .60, probability of failure = .40) or as probability distributions (probability of loss of x ships = F(x)). Either of these may be revised with new information (at least approximately) using Bayes's law (see Ref. 41 for a discussion of the required conditions). For example, if the a priori estimated probability of troops massing is known and the message "troops seen in sector x" comes in, and the conditional probability of (massing/troops seen) is known, then the revised probability of troops massing is:

$$P(\text{massing} \mid \text{troops seen}) = \frac{P(\text{massing}) \cdot P(\text{troops seen} \mid \text{massing})}{P(\text{troops seen})}$$

More general schemes for updating propositions use probability intervals instead of values, relying on the Dempster-Shafer theory of evidence.[42] These approaches model the amount of support and denial assigned to each proposition and have mechanisms for handling inconsistent data.

The uncertainties with respect to prediction of Blue response and prediction of engagement outcome cascade together. The outcome prediction assumes some Blue response and thus is a conditional probability. We can approximately derive the probability of each Blue response from its goodness-of-fit to the situational conditions (see Red Prediction of Blue Responses). The probability of each possible outcome is an output of the Force model. We can then calculate an expected value for a Red action by summing the products of the Blue response probabilities and their expected outcomes, multiplied by the value of the outcome:

$$EV(\text{action j}) = \sum_k \sum_m P(\text{Blue response k} \mid \text{action j}) \cdot P(\text{outcome m} \mid \text{k}) \cdot \text{Value(outcome m)}$$

Some special problems arise with respect to uncertainties with time. Action or event times may be expressed as ranges, within some confidence level (see Ref. 43 for a discussion of this problem). During look-ahead over several moves, such ranges will have to be linked to give a worst case estimate. Also, matching of situational conditions to time criteria becomes probabilistic when time ranges are used. For example, assume that the activating condition for an action is that troops must be in position before 0700 and the projection says the troops will be in position between 0100 and 0900, then the probability of satisfaction of the condition (assuming uniform probability along the time range) would be .75. We expect that a fitting representation of such temporal uncertainties in combination with the other uncertainties present in strategic analysis will be an extremely complex problem, well beyond the scope of this report. Much of the work for the final Mark III system will deal with this problem.

Appendix C

GOODNESS-OF-FIT CALCULATIONS

Section III described the use of essential and optional conditions for deciding on scripts, choosing between branches, and performing continuation tests. Recapping briefly, the essential and optional conditions act quite differently for these tests. The essential conditions act as filters, determining those options that satisfy all necessary conditions, which are specified in a binary yes/no fashion; they are either achieved or not. The optional conditions, however, are invoked if several candidate options satisfy the essential conditions. The optional conditions contribute to the value of an option and so should be interval scaled and weighted in importance. Both goals and costs may be aggregated in this fashion by changing the sign of the costs. The weighted aggregate value is then used to select the best option:

$$\text{Value (option j)} = \sum_i w_i A_{ij}$$

where w_i is the importance weight of attribute i and A_{ij} is the level of attribute i present with option j. (Note that both the weights and attribute levels are normalized to the 0 − 1 range.)

This model assumes that the attributes are independent and their effects are additive. Even if these assumptions do not strictly hold, the additive formulation is generally robust to discrepancies.[44] A more general multiplicative model is also applicable, but the relationship between the factors becomes less transparent than with the additive formulation.

Some of the optional conditions may simply be extensions of the essential conditions. An essential condition for an attack option might be availability of at least three MRDs within 100 km of the line. The optional measure might measure the additional advantage of three to six MRDs, with three MRDs having no additional effect and six MRDs having an additional effect of 1.0.

The aggregate value of the optional conditions may be used to estimate the goodness-of-fit of the possible Blue responses and in turn to estimate the response probabilities. The probability of any response R could be calculated as:

$$P(R_k) = \frac{V(R_k)}{\sum_m V(R_m)}$$

where $V(R_k)$ is the aggregate value of response R_k and R_m represents all possible responses.

Appendix D

SCENARIO FOR A THEATER CONFLICT IN THE PERSIAN GULF

This example scenario (referred to throughout Sec. III) concerns a Red invasion and occupation of Iran with a deliberate phased commitment of forces. The scenario, drawn from Winnefeld and Levine,[45] is intentionally simplified to illustrate specific functions of the Red Agent. Throughout the conflict, both the Red and Blue Agents are assumed to be risk-averse, and the emphasis is on Soviet territorial acquisition options in the Persian Gulf. All of the analysis is in look-ahead.

The situation begins with the following conditions:

- Ample Red forces are available in the area.
- Intelligence shows the political and military status of Iran to be in turmoil.
- Blue land and naval forces in SWAsia are below critical defense levels.

Considering the emphasis on the Persian Gulf area, Red has four script choices, as shown in Fig. D.1.

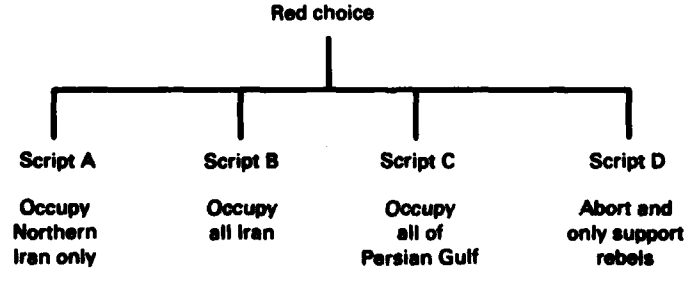


Fig. D.1—Red script options at beginning of scenario

The Red Agent's first step is to check the essential conditions for each candidate script. The Red NCL immediately prunes down the options because several essential conditions for scripts C and D are not satisfied at D − 30.[1] Script C fails as it requires greater Soviet forces than are currently available, along with a reduction in Blue RDF capabilities. Script D fails as it would be invoked only if the Iranian government is perceived to be stable, much larger Blue forces are in place in SWAsia, or NATO commits to participation in the conflict. All of the essential conditions for Scripts A and B, however, are met (at least those that can be checked at D − 30). Red must assume some future conditions, such as the following for Script B, to be true and tag them for later testing during the look-ahead:

- No Blue commitment of forces before D − 13.
- No military role by Turkey before D − 1.
- Red Control of Northern Iran by D + 8.

The Red Agent then makes the choice between scripts A and B on the basis of their optional conditions:

Script A:

1. Blue naval forces between three and four carrier battle groups at D − 30.
2. Iranian military resistance between D-day and D + 10 will be moderate.

Script B:

1. Blue tactical air forces in SWAsia are below two tactical fighter wings.
2. Iranian military resistance between D-day and D + 10 will be light.
3. Blue Naval forces now between one and two carrier battle groups.

Again, we must defer some of the conditions for later testing. Assuming these to be provisionally true, Script B receives a higher evaluation (see Appendix C for the specific form of the goodness-of-fit calculations). Red selects script B to follow and tags script A as the first alternative.

The NCL sends the appropriate actions, time-tagged goals, and allocation of forces to each area command. The script lists which theaters are operational in this type of conflict (Eastern European and Southwest Asia), specific goals for each theater, theater priorities, the actions that pertain to each theater, necessary force levels for the

[1]For brevity, only the failing conditions are noted here.

campaign, and the current distribution of forces in and adjacent to each theater.

Once the resources are allocated, the two ACLs begin at the initial time (D − 30) and trace out a scenario in look-ahead. Script B calls for several actions at D − 30. The SWAsia command must mobilize and deploy its force in the Transcaucasus and Turkestan regions. The Eastern European command must alert its Warsaw Pact forces in the European theater. The NCL must send messages to NATO and Blue about beginning a scheduled military exercise. Each action has slots for refinement. The SWAsia command, for instance, must decide how many troops to mobilize and deploy. The NCL must similarly calculate when to send the messages to NATO and Blue. Filling these slots involves the use of production rules associated directly with the action:

> Force Specification:
>
> If the roads are open in the Transcaucasus and Iranian forces are below 2 divisions in Khorasan, then deploy 3 MRDs in the area, otherwise deploy 2 MRDs.
>
> Time Calculation:
>
> If the deployment of Red forces in the Transcaucasus is completed, wait 2 days and send messages to NATO and Blue describing Red's actions as scheduled military maneuvers.

Once all the slots for an action are filled, the ACL sends the action to the Force Agent for simulation. The Force Agent returns the expected outcomes and the changed situation estimate to the Red world model. Red updates its model and moves look-ahead time forward to the next action.

The Red Agent assumes Blue to be risk-averse and following a defensive script using conventional weapons. Red checks the conditions for actions within this Blue script. The Red mobilization and deployment at D − 28 matches the conditions associated with a set of Blue actions. Red's model of Blue initiates the actions, mobilizing and deploying its forces at D − 28 and sending messages to a host of other countries (Spain, France, Portugal, Egypt, etc.) requesting basing privileges. The responses to these messages will determine the promptness and extent of Blue opposition to Red.

From D − 27 to D − 14, Blue and Red continue their deployments and send additional messages to each other and to their allies. The messages are sent to Scenario, which in turn generates messages and takes actions in response to the projected situation. Scenario sends the

messages to Blue and Red and sends the actions to Force Operations. No branch points or continuation tests are found before D − 14.

At D − 14 (or before, if enough countries send early responses to the basing requests), Blue will choose among three branches: (1) minimum response, (2) phased mobilization and deployment of the RDF (rapid deployment force), and (3) immediate and strong RDF deployment. The first branch would be chosen if all Blue allies refuse basing or do not answer the message. The second branch would be chosen if some countries—such as Spain, Portugal, and Egypt—grant basing privileges or if Red progress has been slowed. The third branch is chosen if all countries give basing permission, allowing a prompt Rapid Deployment Force (RDF) response. For this example, let us assume that Blue chooses the phased mobilization option.

At D − 13 the move returns to Red. Red now has several branch choices. The Red first choice is a rapid invasion, usable only if Blue does not receive basing rights in time and does not quickly deploy the RDF. The Red second choice is a three-phase deliberate commitment of forces, beginning with a rapid advance in Northern Iran with subsequent consolidation over that area, and a final series of attacks along the Zagros. This option is used if Red expects Blue opposition to be moderate, and it has a flexible timetable of 30 to 60 days for control. The final Red option is a slower timetable version of the second branch, used if a strong anti-Red coalition exists. A continuation condition is also present at D − 13:

> If there is a NATO commitment to the conflict, terminate the B script and give control back to the NCL (presumably the NCL will revert to the fall-back D script).

Assuming Blue has chosen its second branch option and intelligence does not indicate a NATO commitment, the Red second branch option is favored. Both Red and Blue have several specified actions from D − 13 to D − 1, including Red halting some of the Red mobilization as a false signal to Blue and Blue deploying in the Persian Gulf and Egypt. Again, all of these actions are refined by filling in slots. After simulation by the Force program, Red enters the projected outcomes into its world model.

By D − 1, many specific Red objectives and situational conditions must be met. Blue deployments must be limited in scope, Red forces must be in position, and intelligence should indicate no escalation or NATO involvement. This appears to be a good point for evaluation. There has been an exchange of actions, there are many time-tagged goals (and situational conditions) to test, and previous choices may be altered. The tests to be made include the following:

- There is no Blue deployment in Iran by D − 1.
- Iran is in military and political turmoil at D − 1.
- There is no NATO mobilization.
- Blue and allies have fewer than six brigades and five tactical fighter wings in SWAsia at D − 1.
- Red Forces are in position in the Transcaucasus and Turkestan areas by D − 3.

If any of these conditions or goals are not met, a table look-up specific to the script is used to determine which action to modify. The table entries are of the following type:

> If less than x Red MRDs are in position in the Transcaucasus, modify the mobilize-and-deploy action at D − 30.

The mobilize-and-deploy action might have back-up slots of mobilizing sooner (at D − 32) and of mobilizing a larger force. The SWAsia command tries each of these in turn, and reinitiates the process of look-ahead and action selection at that point. If the ACL exhausts all of the backup actions in this way and still does not satisfy the essential goals, it alerts the NCL. The NCL attempts a different resource allocation and if this fails reverts to the A script, as it is tagged as the first alternative.

The move analysis ends at this point because sufficient evidence is present to support or deny the Red script choice and specify the initial set of Red actions. No further look-ahead is necessary. If the full set of objectives is met, the B script is chosen for invocation. If some objectives are unsatisfied and other scripts have not yet been investigated (such as the A script), that script is examined next. Once the NCL chooses a script for invocation, it requests the Systems Monitor to advance the simulation clock and perform the series of Red action choices up to the Blue move. Following the Blue move, Red performs its entire analysis again, unless Blue makes the precise move Red predicted and the expected outcome occurs.

REFERENCES

1. P. Davis and J. Winnefeld, *The Rand Strategy Assessment Center: An Overview and Interim Conclusions About Utility and Development Options*, The Rand Corporation, R-2945-DNA, March 1983.
2. W. Schwabe and L. Jamison, *Scenario Agent: A Rule-Based Policy-Level Model of Non-Superpower Behavior in Strategic Conflicts*, The Rand Corporation, R-2926-DNA, September 1982.
3. B. Bennett, C. Williams, and A. Bullock, *Conceptual Design of an Advanced Force Agent for the Rand Strategy Assessment Center*, The Rand Corporation, unpublished briefing notes.
4. W. Jones, J. LaCasse, and M. LaCasse, *The Mark II Red and Blue Agent Control Systems for the Rand Strategy Assessment Center*, The Rand Corporation, N-1838-DNA, April 1983.
5. J. Carbonell, *Subjective Understanding: Computer Models of Belief Systems*, University of Michigan Research Press, Ann Arbor, 1981.
6. J. Carbonell, "POLITICS: Automated Ideological Reasoning," *Cognitive Science*, 2, 1978, pp. 27-51.
7. R. Schank and R. Abelson, *Scripts, Plans, Goals and Understanding*, Lawrence Erlbaum, Associates, Hillsdale, N.J., 1977.
8. N. Nilsson, *Problem-Solving Methods in Artificial Intelligence*, McGraw-Hill, New York, 1971.
9. J. Pearl, "Heuristic Search Theory: Survey of Recent Results," *Proceedings IJCAI-81*, Vancouver, pp. 554-562.
10. F. Hayes-Roth, "The Role of Partial and Best Matches in Knowledge Systems," in F. Hayes-Roth and D. Waterman (eds.), *Pattern Directed Inference Systems*, Academic Press, New York, 1978.
11. J. Barnett and M. Bernstein, *Knowledge-Based Systems: A Tutorial*, System Development Corp., Technical Report TM-(L)-5903/000/00, Santa Monica, June 1977.
12. R. Wesson, "Problem-Solving with Simulation in the World of an Air Traffic Controller," Doctoral Dissertation, University of Texas at Austin, 1977.
13. L. Miller, M. Merkhofer, R. Howard, J. Matheson, and T. Rice, *Development of Automated Aids for Decision Analysis*, Technical Report, Stanford Research Institute, Menlo Park, Calif., May 1976.

14. R. Brown, C. Hoblitzel, C. Peterson, and J. Ulvila, *Decision Analysis as an Element in an Operational Decision Aiding System*, Technical Report 74-2, Decisions and Designs, Inc., McLean, Virginia, September 1974.
15. R. Duda and P. Hart, *Pattern Classification and Scene Analysis*, Wiley-Interscience, New York, 1973.
16. R. Davis, B. Buchanan, and E. Shortliffe, "Production Rules as a Representation for a Knowledge-Based Consultation Program," *Journal of Artificial Intelligence*, Vol. 8, 1977, pp. 15-45.
17. B. Buchanan, G. Sutherland, and E. Feigenbaum, "Heuristic Dendral: A Program for Generating Explanatory Hypotheses in Organic Chemistry," *Machine Intelligence 4*, American Elsevier, New York, 1969.
18. L. Erman, F. Hayes-Roth, V. Lesser, and D. Reddy, *The Hearsay-II Speech Understanding System: Integrating Knowledge to Resolve Uncertainty*, Carnegie-Mellon University Technical Report, CMU-CS-79-156, Pittsburgh, 1980.
19. J. Battilega, *U.S./Soviet Asymmetries in Strategic and Theater Nuclear Warfare*, presentation to the 48th MORS, December 1981.
20. J. Erickson and E. Feuchtwanger, *Soviet Military Power and Performance*, Archon Books, Hamden, Conn., 1979.
21. R. Wesson and F. Hayes-Roth, *Dynamic Planning: Searching Through Time and Space*, The Rand Corporation, P-6266, February 1979.
22. A. Babich, J. Grason, and D. Parnas, "Significant Event Simulation," *Communications of the ACM 18(6)*, 1975, pp. 323-329.
23. R. Levine and J. Winnefeld, *Campaign Analysis and Development of Analytic War Plans*, The Rand Corporation, N-1836-DNA (forthcoming).
24. J. Fain, D. Gorlin, F. Hayes-Roth, S. Rosenschein, H. Sowizral, and D. Waterman, *The Rosie Language Reference Manual*, The Rand Corporation, N-1647-ARPA, December 1981.
25. R. Steeb and S. Johnston, "An Interactive System for Group Decisionmaking," *IEEE Proceedings on Systems, Man and Cybernetics*, August 1981.
26. J. Carbonell, "A Computational Model of Analogic Problem Solving," *Proceedings IJCAI-81*, Vancouver, pp. 147-152.
27. A. Newell and H. Simon, *Human Problem Solving*, Prentice-Hall, Englewood Cliffs, New Jersey, 1972.
28. J. Becker, "A Model for the Encoding of Experimental Information," in R. Shank and K. Colby (eds.), *Computer Models of Thought and Language*, 1973, pp. 396-446.

29. W. Kornfield, "The Use of Parallelism to Implement a Heuristic Search," *Proceedings IJCAI-81,* Vancouver, pp. 575-578.
30. D. Waterman and F. Hayes-Roth, "An Overview of Pattern-Directed Inference Systems," in D. Waterman and F. Hayes-Roth (eds.), *Pattern-Directed Inference Systems,* Academic Press, New York, 1978.
31. B. Hayes-Roth, F. Hayes-Roth, S. Rosenschein, and S. Cammarata, "Modeling Planning as an Incremental, Opportunistic Process," *Proceedings IJCAI-79,* Kyoto, Japan, 1979, pp. 375-383.
32. M. Fox, "Reasoning with Incomplete Knowledge in a Resource-limited Environment: Integrating Reasoning and Knowledge Acquisition," *Proceedings IJCAI-81,* Vancouver, pp. 313-318.
33. M. Samet and K. Davis, *Computer-Based Supervisory System for Managing Information Flow in C3 Systems,* Perceptronics, Inc., Woodland Hills, Calif., Technical Report PTR-1033-77-3, March 1977.
34. A. Madni, R. Steeb, D. Rubio, and R. Farnum, *Adaptive Information Selection for Support of C-3 Functions in Small Tactical Units,* Perceptronics, Inc., Woodland Hills, Calif., Technical Report PFTR-1061-79-1, January 1979.
35. T. Garvey, J. Lowrance, and M. Fischler, "An Inference Technique for Integrating Knowledge from Disparate Sources," *Proceedings IJCAI-81,* Vancouver, pp. 319-325.
36. R. Duda, P. Hart, N. Nilsson, and G. Sutherland, "Semantic Network Representations in Rule-Based Inference Systems," in F. Hayes-Roth and D. Waterman (eds.), *Pattern-Directed Inference Systems,* Academic Press, New York, 1978.
37. C. Hewitt, "Viewing Control Structure as Patterns of Message Passing," *Artificial Intelligence,* 8, 1977, pp. 323-363.
38. A. Kay, "A Personal Computer for Children of All Ages," *Proceedings of the ACM National Conference,* August 1972.
39. K. Kahn, "Director Guide," MIT AI Lab, Memo 482, June 1978.
40. D. MacArthur and H. Sowizral, "An Object-Oriented Language for Constructing Simulations," *Proceedings IJCAI-81,* Vancouver, 1981.
41. E. Pednault, S. Zucker, and L. Muresan, "On the Independence Assumption Underlying Subjective Bayesian Updating," *Artificial Intelligence,* 16(2), May 1981, pp. 213-222.
42. R. Quinlin, "Consistency and Plausible Inference," The Rand Corporation, P-6831, October 1982.
43. J. Allen, *Maintaining Knowledge About Temporal Intervals,* Computer Science Department Report TR-86, University of Rochester, January 1981.

44. R. Keeney and H. Raiffa, *Decision Analysis with Multiple Conflicting Objectives, Preference, and Value Trade-offs,* Wiley, New York, 1975.
45. J. Winnefeld and R. Levine, *Illustrative Experiments with an Interim Version of Rand's Strategy Assessment Center,* The Rand Corporation, N-1917-DNA, November 1982, For Official Use Only.

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ DISTRUC... BEFORE COMPLETI...

1. REPORT NUMBER: R-2977-DNA
2. GOVT ACCESSION NO.:
3. RECIPIENT'S CATALOG NUMBER:

4. TITLE (and Subtitle): Design for an Advanced Red Agent for the Rand Strategy Assessment Center
5. TYPE OF REPORT & PERIOD COVERED: interim
6. PERFORMING ORG. REPORT NUMBER:

7. AUTHOR(s): Randell Steeb and James Gillogly
8. CONTRACT OR GRANT NUMBER(s): DNA001-80-C-0298

9. PERFORMING ORGANIZATION NAME AND ADDRESS:
The Rand Corporation
1700 Main Street
Santa Monica, CA 90406
10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS:

11. CONTROLLING OFFICE NAME AND ADDRESS:
Director
Defense Nuclear Agency
Washington, D.C. 20305
12. REPORT DATE: May 1983
13. NUMBER OF PAGES: 72

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office):
Director of Net Assessment
Office of the Secretary of Defense
Washington, D.C. 20301
15. SECURITY CLASS. (of this report): unclassified
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE:

16. DISTRIBUTION STATEMENT (of this Report):
Approved for Public Release; Distribution Unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report):
No restrictions

18. SUPPLEMENTARY NOTES:

19. KEY WORDS (Continue on reverse side if necessary and identify by block number):
USSR
War Games
Strategic Analysis
Artificial Intelligence

20. ABSTRACT (Continue on reverse side if necessary and identify by block number):
This report describes Rand's proposed approach for building an advanced "Red Agent" computer model to describe Soviet behavior in the context of an automated war game. The advanced (Mark III) Red Agent described in this report will use and extend state-of-the-art artificial intelligence techniques to produce an extremely flexible model able to reflect the best-estimate and contrary-view concepts on likely Soviet political-military behavior in major superpower conflicts.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

END

DATE
FILMED

11 83

DTIC